RELATÓRIO DE PROJETO DE APTIDÃO PROFISSIONAL

LINE LIGHT

PEDRO ESPANHA TORRADO DA SILVA
ALUNO Nº: 2219063

PROFESSOR ORIENTADOR: ANTÓNIO BRANCO

TURMA DE GESTÃO DE EQUIPAMENTOS INFORMÁTICOS
(TGEI_19)

1. Índice

1. Índice 2
2. Introdução 3
3. Objetivos de Sustentabilidade ONU 4
4. Calendarização do Projeto 5
5. Funcionalidades do Dispositivo 6
6. Componentes Necessários e Custos do Desenvolvimento 8
7. Circuitos a ser utilizados 10
7.1 Comando 10
7.2 Placa Central 13
7.3 LEDs 14
8. Programa 18
8.1 Comando 18
8.2 Placa Central 25
9. PCBs 28
9.1 Comando 28
9.2 Placa Central 30
9.3 LEDs 31
10. Modelos 3D 32
10.1 Comando 32
10.2 Placa Central 37
10.3 Piscas 40
10.4 Fivela 42
11. Implementação 44
12. Componentes 46
13. Ficheiros em Anexo 48
14. Índice de Imagens 49

2. Introdução

Ao longo deste trabalho tive como objetivo desenvolver um produto que tenha como função facilitar e tornar mais segura a utilização de veículos sem sinais luminosos.
Grande parte da segurança rodoviária baseia-se na comunicação entre condutores dos diferentes veículos de forma que qualquer possível interação ocorra da maneira mais segura possível e que se possa evitar qualquer mal-entendido, criando assim um ambiente em que todos os condutores se sintam seguros.
Contudo, no nosso dia-a-dia, podemos encontrar muitos meios de transporte que coexistem na via pública e que não possuem elementos luminosos de sinalização. Em vez disso, estes condutores utilizam sinais gestuais caso tenham a intenção de alterar de via, realizar uma mudança de direção ou diminuir a velocidade do veículo, mas estes, obviamente, apresentam vários problemas quando comparados com sinais luminosos, principalmente em situações de pouca luminosidade do ambiente, como condução durante a noite ou durante o atravessamento de uma passagem inferior.
Como solução para este problema decidi começar a desenvolver o produto a que chamei de *Line Light*.
Começando pelo nome, decidi chamar ao produto *Line Light* dada a sua função, sendo esta possibilitar a utilizadores de meios de transporte que não possuem sinais luminosos (*Light* significando Luz em inglês) a utilização dos mesmos de forma a indicar o trajeto para onde o sujeito se destina (*Line* tendo o significado de linha, fila ou reta em inglês).

3. Objetivos de Sustentabilidade ONU

Este projeto foi desenvolvido tendo em consideração um conjunto de objetivos de desenvolvimento sustentável (ODSs).

Figura 1 - Objetivos de Desenvolvimento Sustentável

Os Objetivos de Desenvolvimento Sustentáveis são compostos por 17 objetivos e 169 metas que a ONU (Organização das Nações Unidas) pretende atingir até 2030 e estas foram criadas durante a Cúpula das Nações Unidas sobre o Desenvolvimento Sustentável em setembro de 2015 (https://news.un.org/pt/tags/cupula-das-nacoes-unidas-para-sobre-o-desenvolvimento-sustentavel).

O meu projeto engloba os segintes ODS:

03 – "Saúde e bem-estar: assegurar uma vida saudável e promover o bem-estar para todos, em todas as idades." Uma vez que tem como objetivo evitar acidentes rodoviários e assim preservar a saúde e bem-estar de qualquer utilizador de veículos sem indicadores luminosos e também das pessoas que o rodeiam.
09 – "Inovação infraestrutura: construir infraestrutura resiliente, promover a industrialização inclusiva e sustentável, e fomentar a inovação." Todo o projeto é composto pelo desenvolvimento de novos produtos que funcionam à base de baterias, de forma sustentável, sendo possível utilizar materiais sustentáveis para a produção de alguns dos componentes.
12 – "Consumo e produção responsáveis: assegurar padrões de produção e de consumo sustentáveis." O projeto foi desenvolvido tendo em atenção que teria que ser de fácil reparação, sendo assim possível reparar o dispositivo sem contribuir para o aumento dos resíduos eletrônicos.

4. Calendarização do Projeto

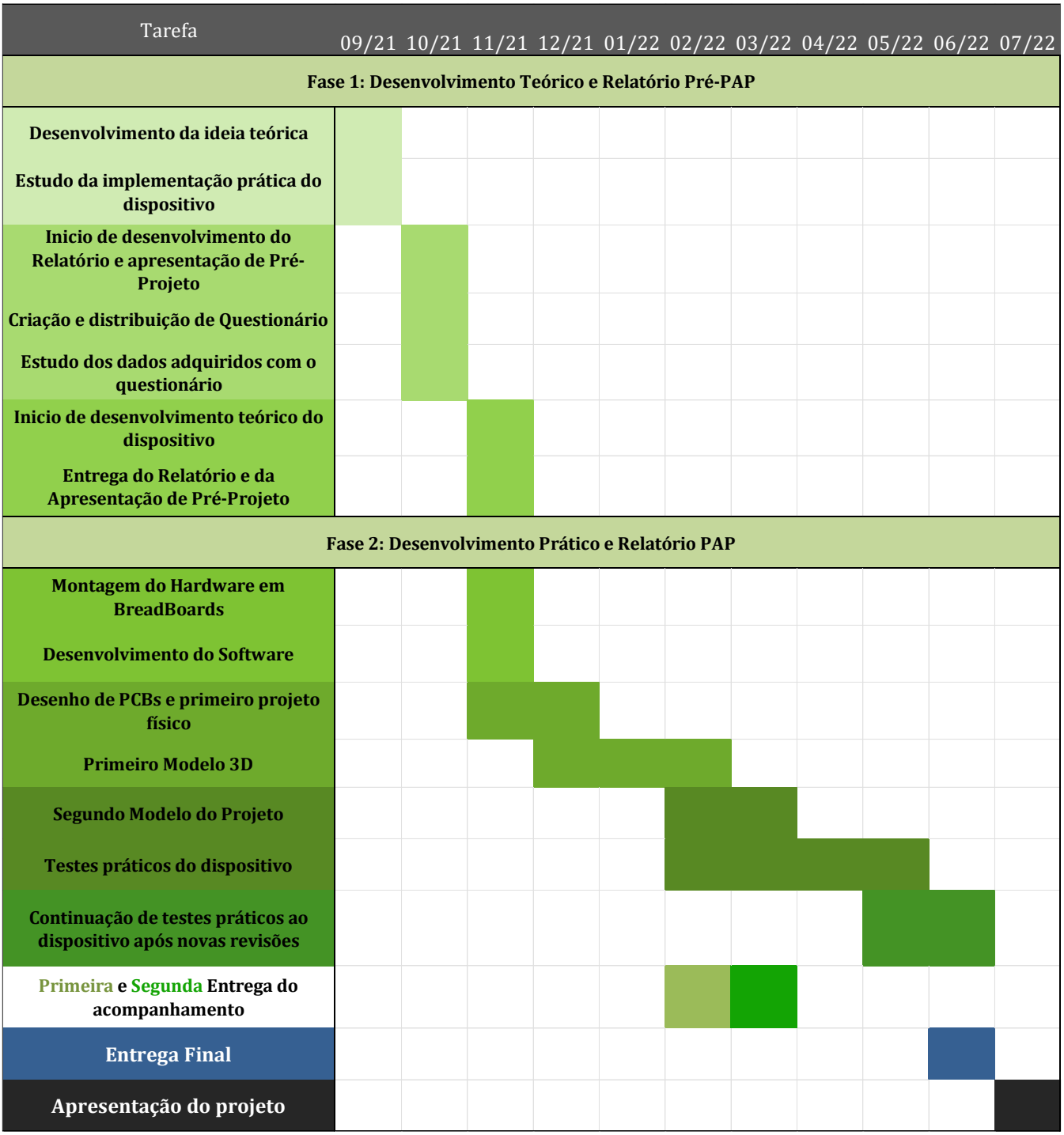

Figura 2 - Cronograma

A Calendarização pode ser verificada no ficheiro 2, referido nos ficheiros em anexo.
Algumas destas datas não coincidiram com as datas planeadas inicialmente, uma vez que foi criada uma segunda versão do dispositivo após testes iniciais.

5. Funcionalidades do Dispositivo

Ao planear este projeto comecei por refletir se este projeto era interessante para outros possíveis utilizadores e para isso decidi criar um questionário, com perguntas relacionadas com o possível interesse por este tipo de dispositivo, e o tipo de funcionalidades que poderia introduzir chegando à conclusão de que havia um conjunto de funcionalidades que queria que fizessem parte do dispositivo, sendo essas:

1. Ter a capacidade de realizar sinalização luminosa antes do utilizador realizar uma mudança de direção ou de via.
2. Ser um dispositivo compacto e portátil, mesmo quando não está em uso;
3. Ser um dispositivo versátil de forma a poder ser utilizado com um conjunto de meios de transporte;
4. Ser um dispositivo de utilização intuitiva e natural;
5. Ser um dispositivo seguro e que seja capaz de avisar o utilizador caso haja algum erro e o seu sinal não seja transmitido da forma pretendida;
6. Ser um dispositivo que tenha resistência a condições climatéricas exteriores adversas e longa durabilidade;
7. Ser um objeto estético e discreto;
8. Ser possível a reparação sem ser necessário substituir todo o dispositivo.

De forma a ir ao encontro destes requisitos decidi utilizar dois dispositivos separados, um comando e um conjunto central (composto por uma placa central ligada aos 2 piscas) com comunicação sem fios entre si, de forma a satisfazer os pontos 2, 3, 7. De forma a ter em conta o ponto 5, é necessário que estes dois dispositivos possam comunicar entre si para que possam fazer o seguinte sistema de verificação.

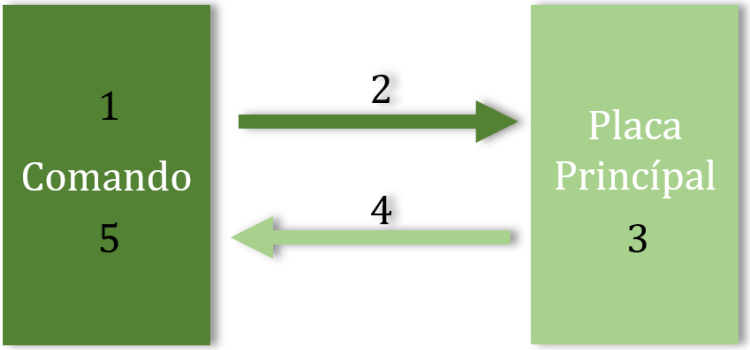

Figura 3 - Sistema de verificação

1 – Uma ação do utilizador gera um sinal;
2 – Esse sinal é transmitido para a placa central;
3 – A placa central produz a ação pretendida pelo utilizador;
4 – Um sinal é emitido pela placa central para o Comando;
5 – O comando produz uma ação que demostra ao utilizador que a sua ação gerou a reação pretendida pela placa central.

Sendo este dispositivo sem fios, tive que realizar diversos testes com várias formas de comunicação sem fios, entre as quais:
Radiofrequência – Não era possível comunicação bidirecional a não ser que fossem utilizados 4 módulos, um recetor e um emissor por dispositivo. Para testes utilizei dois módulos de 433MHz (Componente 1.
Infravermelhos – Para além de ter o mesmo problema da Radiofrequência, estes têm um sinal muito direcionado e que não pode ser obstruído uma vez que este sinal é enviado por sinais luminosos (Componente 2 e 3).
A última opção, que testei e aquela que acabou por ser utilizada foram módulos Bluetooth HC–05 e HC-06 (Componente 4). Uma vez que a partir de Bluetooth é possível ter comunicação bidirecional e sendo estes Bluetooth v2.1 estes têm um alcance registado de 100 metros. Para além disso, escolhi estes módulos pois eles são os indicados para utilização SPP (Serial Port Protocol) o que facilita bastante a sua implementação neste projeto. Uma vez que o dispositivo funciona por Bluetooth este poderia também funcionar a partir de conexão ao telemóvel o que é possível e bastante fácil de conectar, desde que se utilizem aplicações que também utilizam o protocolo SPP. Assim é possível também desenvolver uma aplicação especificamente para este propósito, apenas não foi feito por não ser o foco deste projeto.

Considerando o ponto 6, é necessário ter o dispositivo o mais isolado o possível (tendo em conta a funcionalidade) do exterior, para isso decidi que a melhor forma de carregar o dispositivo é por Indução Eletromagnética. Apesar de ter criado a possibilidade de o fazer a partir de contactos elétricos, esses acabaram por não ser utilizados uma vez que era difícil isolar os mesmos do interior da caixa do dispositivo, mas com melhores ferramentas de produção seria possível e PCB tem essa capacidade de expansão num modelo futuro.

Tal como o carregamento, é necessário que a interação com o comando se encontre isolada do exterior. Este foi um dos maiores problemas que encontrei quando estava a projetar o funcionamento do mesmo. Poderia ser feito de várias formas, mas grande parte envolvia desenhos CAD demasiado desenvolvidos e com tolerâncias demasiado apertadas para as minhas competências e para as ferramentas que tenho à minha disposição. Assim optei pela utilização de sensores de campo electromagnéticos (Hall Effect Sensor) e Ímanes de Neodímio. Este método pouco convencional torna possível a existência de uma parede entre o dispositivo e a forma de interação com o mesmo.

6. Componentes Necessários e Custos do Desenvolvimento

De forma a organizar os componentes em questão, dividi-os em tabelas:

Componentes de teste e produção	***Quantidade***	***Número de Componente***
Breadboard	3	6
Arduino Nano	2	7
Cabo Usb 2.0 mini B	2	8
Carregador de baterias de lítio tp4056	2	9
Baterias de lítio mr18650 (3.7v e 2200mAh)	2	10
Cabos de conexão	30	11
Arduino Uno	1	12
FTDI FT232RL	1	13

Comando e placa central	***Quantidade***	***Número de Componente***
Módulo Bluetooth HC-05 (configurado em modo 0)	1	4
Módulo Bluetooth HC-06	1	5
LEDs (Díodos emissores de luz)	4	14
Transístores PNP	2	15

Indicadores Luminosos de Direção (2 conjuntos)	***Quantidade***	***Número de Componente***
Temporizador NE555 (NE555 Timer)	2	16
CD4017 Contador de Década (CD4017 Decade Counter)	2	16
Condensadores de diversos valores	4	17
Resistências de diversos valores	10	18
LEDs (Díodos emissores de luz)	10	14
Díodos	1	19

Para calcular o preço de materiais na produção deste projeto, criei um ficheiro BOM (Bill Of Materials ou Lista de Materiais), em anexo, e que será atualizado ao longo deste projeto, de forma a manter o preço de produção sempre atualizado, uma vez que pretendo adicionar vários componentes a esta lista e alterar outros.
As tabelas de preços por componente podem ser reduzidas em tabelas de preço por dispositivo, como a que está apresentada abaixo.
No ficheiro em anexo pode verificar-se que grande parte dos componentes são ficheiros SMD, pois a certa altura no desenvolvimento do projeto apercebi-me que, para certos circuitos e alguns dos dispositivos, era mais prático, sendo que o preço por dispositivo reduziu drasticamente, mas componentes SMD têm que ser comprados em massa, o preço apresentado na tabela abaixo é o preço de cada um dos dispositivos com esses componentes, sendo que a encomenda de todos os componentes SMD (que se podem verificar mais tarde nos circuitos) para um comando ficou exatamente 8.74€ (excluindo portes).

Dispositivos	***Preço (€) de Protótipo***	***Preço (€) em massa***
Placa Central (Main Board)	14.72	5.61
Pisca (Blinker)	1.83	1.83
Pisca (Blinker)	1.83	1.83
Comando (Controller)	16.89	8.21
Total*	35.27	24.48

Contudo, esses valores aplicam-se apenas a um conjunto de componentes e não ao preço de desenvolvimento deste projeto. É difícil dizer exatamente o valor gasto no desenvolvimento deste projeto, considerando os componentes que já tinha em minha posse e à quantidade de material gasto na produção de protótipos. Mas, tendo em consideração os PCBs comprados e os componentes comprados para este projeto. Este fica aproximado do valor de 363.34€. Os valores podem ser verificados no ficheiro "Encomendas" em anexo.

7. Circuitos a ser utilizados

Os circuitos nas imagens seguintes estão divididos por secções. Para auxiliar a verificação dos circuitos na íntegra, que é mais fácil de seguir e compreender, estes encontram-se na pasta "Circuitos".

7.1 Comando

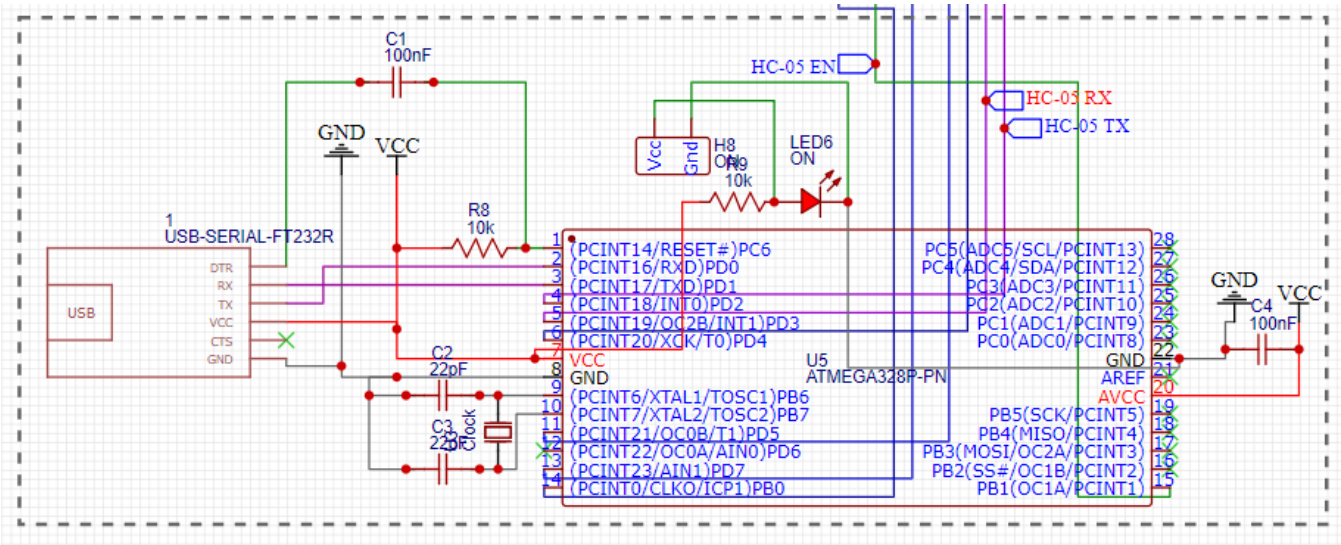

Figura 4 - ATMEGA328P Comando

Começando pelo circuito de ATMEGA328P, este necessita de 2 condensadores de 22pf e um oscilador de cristal de 16Mhz, conectado aos pinos 9 e 10 como representado na figura 4 , de forma que este funcione com um clock externo e assim se possam utilizar funções como millis() e delay(). Além disso, o microprocessador encontra-se conectado a um condensador de 100nF nas entradas a GROUND e VCC, para filtragem de alta frequência e evite variações de tensão no microprocessador. Encontra-se também conectado a uma resistência 10kΩ no pin de RESET conectado a VCC de forma a que este não esteja em RESET constante. O microprocessador está ainda desenhado de forma a poder conectar-se um FT232R, um programador de microprocessadores. Os pinos serão usados para fazer upload do software no microprocessador. O LED indica quando o microprocessador está ligado.

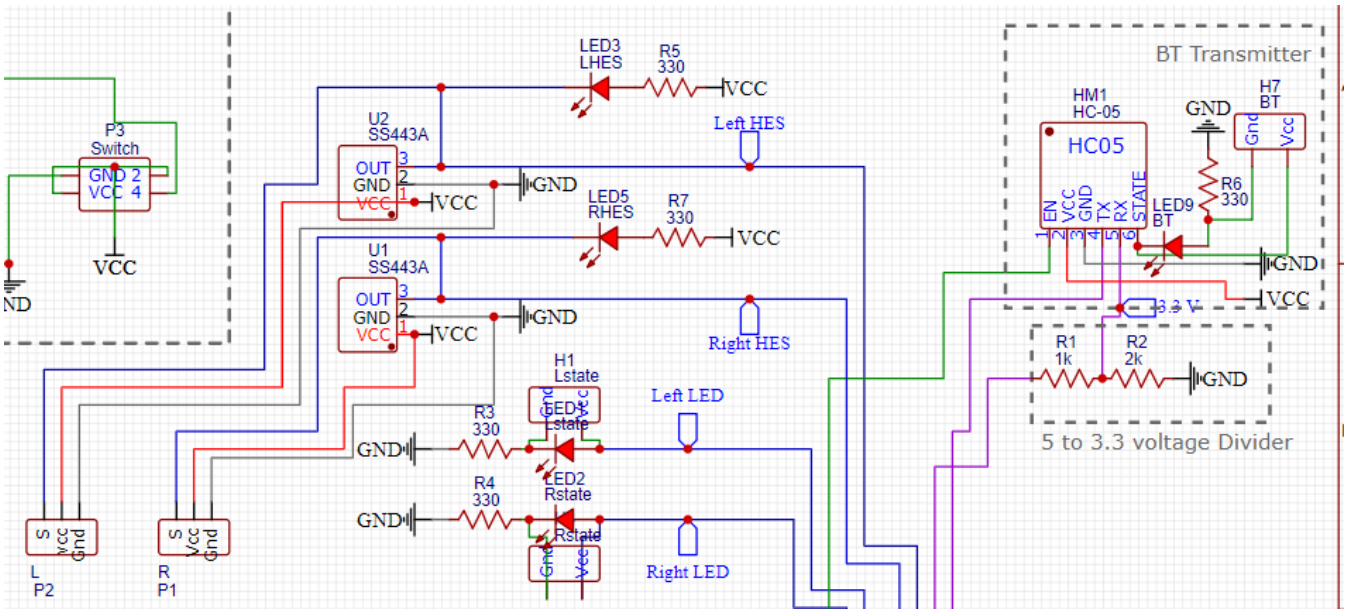

Figura 5 - Circuito de SS443A e HC-05

As saídas do microprocessador, sinalizadas a azul, estão por sua vez ligadas a dois SS443A (Hall Effect Sensors) conforme apresentado na figura 5 e estes têm as suas saídas ligadas a pinos ordenados Signal, VCC e GND, para ser utilizável com módulos a3144 (Componente 20). E o seu output está também ligado a LEDs com as respetivas resistências necessárias para que ao haver um sinal, haja um sinal visível, neste caso a luz destes LEDs.

Se verificarmos o lado direito da imagem acima, podemos ver um módulo HC05 (Programado como MASTER) e com os pinos verdes e roxos conectados, como se pode ver nas duas figuras acima (4 e 5). É importante, ao utilizar um componente como este, que só suporta entrada de sinal a 3.3V, criar um divisor de tensão como o apresentado na figura, podendo assim ligar este a um circuito de 5V sem haver o perigo de estragar o mesmo, principalmente se este estiver a receber sinais mais longos.

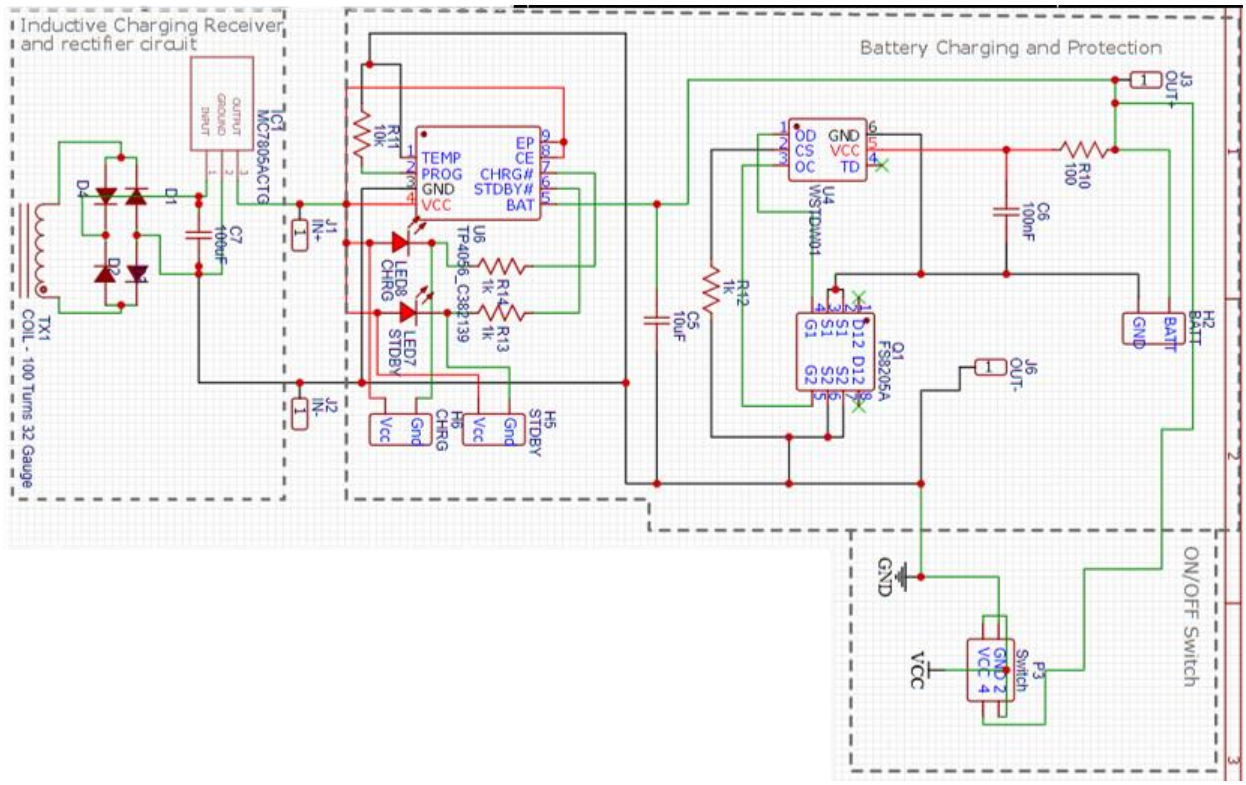

Figura 7 - Circuito de Carregamento

Esta parte do circuito é a parte responsável por carregar a bateria. Se olharmos para o circuito na figura 6 da esquerda para a direita, este começa por uma bobina de 100 voltas de um fio de cobre de 0.27mm, que, na proximidade de um circuito que crie um campo de indução eletromagnética, faz com que haja um input em corrente alternada no circuito. Para tornar esta corrente alternada em corrente contínua é utilizado um conjunto de díodos de forma a criar um retificador de ponte completa e um condensador de 100uF para que a corrente em pulsos do retificador se torne o mais próximo possível de corrente contínua. O funcionamento do rectificador encontra-se representado nas figura 7.

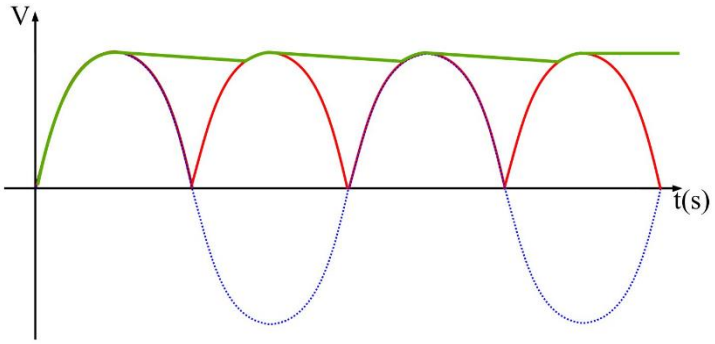

Figura 6 - Corrente Alternada (Azul), Sinal Retificado (Vermelho), Corrente Contínua (Verde)

É preciso ter em atenção que, ao entrar no circuito, a corrente não está a 5V nem a nenhum valor constante, uma vez que esse valor varia conforme o quão próxima a bobina deste circuito se encontra do carregador. De forma a que este valor, agora em corrente contínua, seja

constantemente 5V, é necessário um regulador de tensão (neste caso utilizei um mc7805). Agora a 5V este o circuito passa por um TP4056 (existem vários módulos com este circuito integrado que servem exatamente para esta aplicação e é bastante fácil encontrar online) que é um circuito integrado para carregamento de baterias de Li-ion (Ião de Lítio). Este está conectado, conforme o recomendado na sua ficha de dados, a um circuito fs8205a, que é um MOSFET NPN, e que é utilizado nesta configuração de forma a funcionar como um circuito de proteção da bateria, assim como o circuito integrado WSTDW01 que tem como característica a proteção contra sobrecarga e gera um clock interno, facilitando a sua implementação num circuito como este. De seguida, este circuito termina em 4 pinos, 2 para a saída de corrente, para que o circuito enquanto carrega utilize a corrente diretamente deste circuito, em vez de descarregar a bateria à qual estão ligados os outros 2 pinos. Tudo isto ligado a um botão interruptor que permite ligar e desligar este circuito.

7.2 Placa Central

Na placa central o circuito de carregamento da bateria é exatamente igual ao do comando e o seu circuito de microprocessador também é muito parecido, não havendo, deste modo, necessidade de o demonstrar. A única diferença entre os dois é o que se faz com esses recursos.

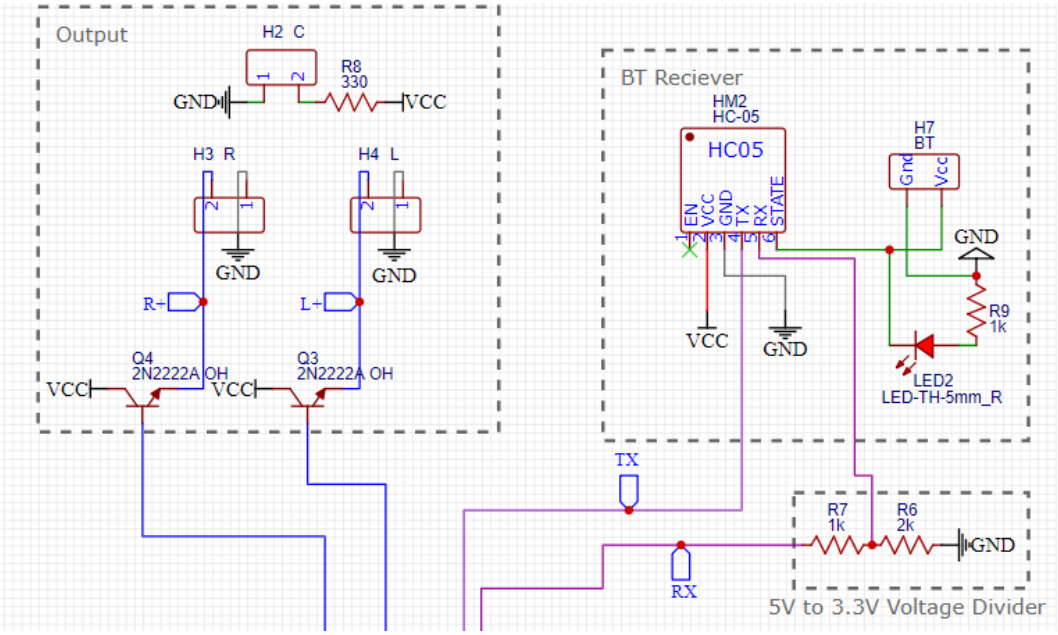

Figura 8 - Circuito Comando

Como podemos verificar na imagem em cima, este circuito também se encontra conectado a um HC05. Contudo, este tem que se encontrar na definição SLAVE para que funcione corretamente, para esse resultado pode também utilizar-se apenas um HC06, uma vez que este não permite a alteração de estado para MASTER e por isso não pode ser utilizado no comando. De resto essa parte do circuito também é idêntica. Sendo apenas diferente a parte de OUTPUT do circuito, o qual é apenas 2 transístores conectados e dois pinos cada para cada um dos circuitos de funcionamento dos LEDS (havendo um terceiro apenas constantemente ligado que pode ser utilizado para testes ou para uma 3ª luz).

7.3 LEDs

Sendo o meu objetivo tornar este produto modular, tentei evitar a utilização de microcontroladores por componente modular, para além dos dois necessários para a comunicação entre o comando e a placa central. Assim, optei por desenvolver os circuitos restantes, sendo estes os circuitos dos indicadores luminosos de direção, apenas em hardware (como se pode verificar na tabela com o mesmo nome apresentada no tópico anterior) e sem qualquer software.
Acrescentando a isso, queria ainda que estes não piscassem apenas (uma vez que estarão próximos um do outro, tendo apenas aproximadamente a largura da bicicleta como superfície de apresentação) mas representassem a direção através de uma animação, deixando assim clara a direção que o utilizador pretende indicar.

Para essa animação decidi utilizar circuito integrado com função oscilador e um contador de década, de forma a criar um circuito com a animação desejada.

Figura 9 - NE555P

Circuito Integrado com função oscilador - NE555P

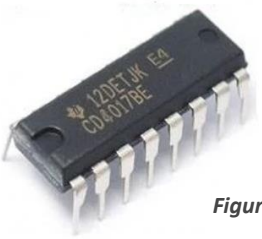

Figura 10 - CD4017BE

Contador de década - CD4017BE

Assim, seguindo as especificações dos circuitos integrados e ficha de dados fornecida pelas empresas fornecedoras de cada um desses produtos, desenvolvi o esquema do circuito obtendo o circuito que irei utilizar e que vou proceder a explicar.

Sabendo os componentes a utilizar, como demonstrados na figura que veremos mais tarde (figura 12), podemos utilizar a fórmula do circuito integrado NE555P, presente nos documentos do mesmo, para calcular a frequência do pulso de sinal do mesmo, sendo essa:

f = 1.44/(R4+2xR2) * C1
f = 1.44/(10k+2*100k) * 2.2uF

E produzindo, assim, um resultado de:

f = 3.123Hz

E sabendo que o período é o inverso da frequência, obtemos:

T = 0.320s

Os restantes valores possíveis de calcular estão organizados no ficheiro 5 em anexo.

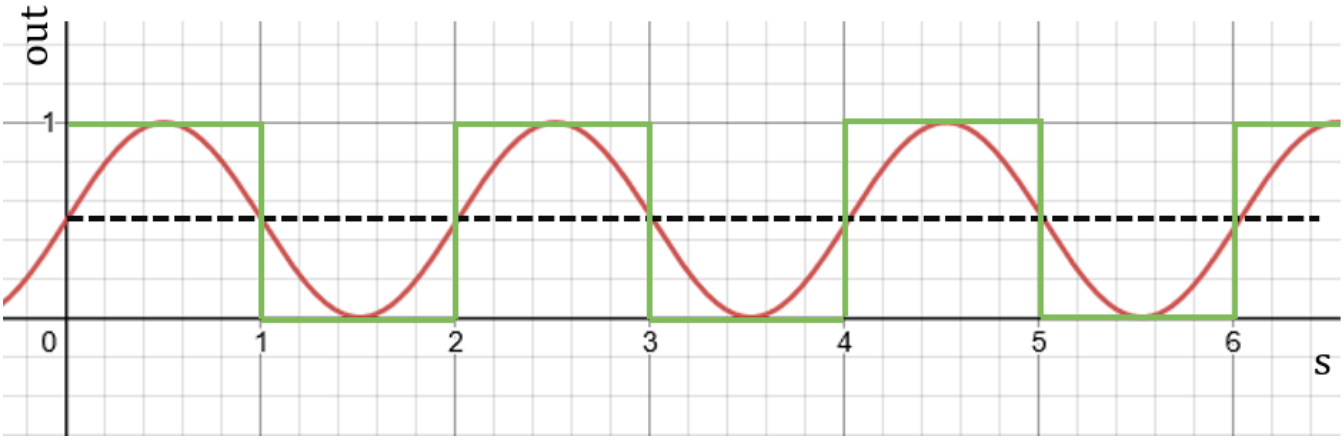

Figura 11 - gráfico de output do circuito integrado NE555P

Este gráfico (figura 12) representa a conversão de um gráfico sinusoidal (representado a vermelho) do circuito integrado num gráfico do output digital (representado a verde) do mesmo circuito no qual o limite é a metade da amplitude da sinusoidal (representada pelo tracejado a preto).

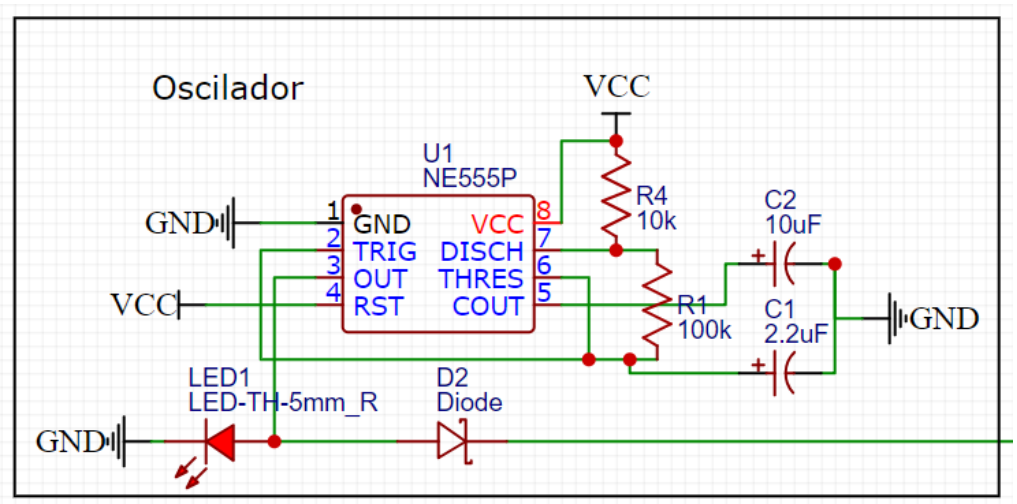

Figura 12 - Circuito de Contador de Década e Output do sistema

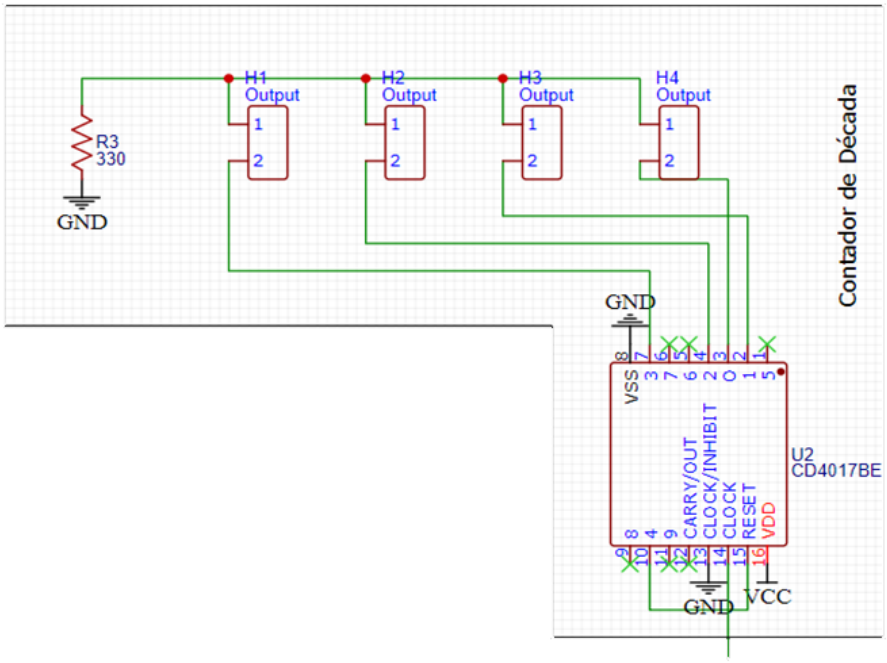

Figura 14 - Circuito com função de Oscilador

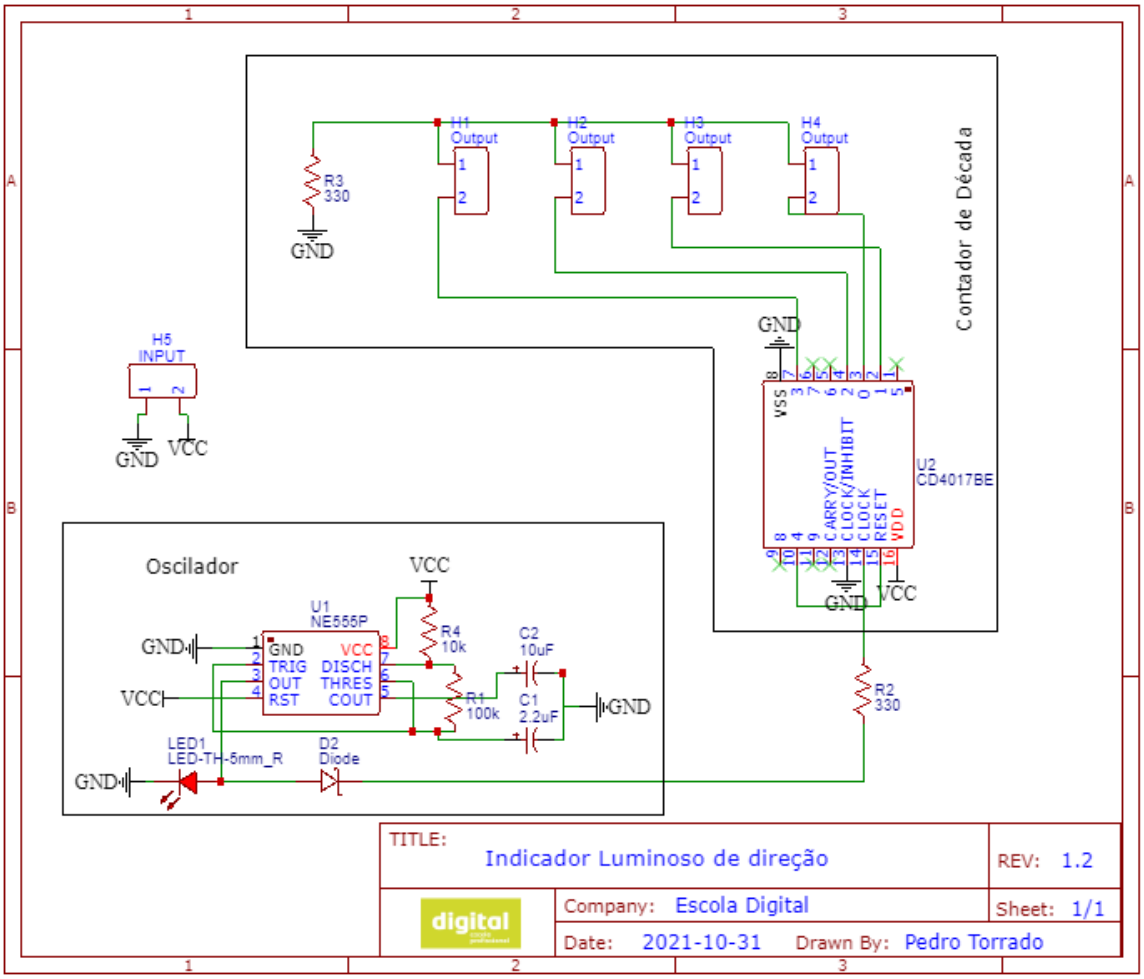

Figura 13 - Circuito completo de Indicador Luminoso de Direção

Este circuito funciona da seguinte forma: O *contador de década* altera o seu pin de *Output* cada vez que recebe um sinal no seu pin de *Clock*, e uma vez que este está conectado ao pin de *Output* de um circuito de *oscilador*, o *contador de década* altera o seu *Output* conforme a frequência definida pelo mesmo (3.123Hz, como definimos anteriormente), criando um ciclo estável que pode ser usado como animação e na qual será aceso uma fita led por cada *Output*, simulando assim o movimento dos leds e descrevendo o movimento que o utilizador pretende realizar. Este circuito não foi o circuito final e tiveram que ser acrescentados transístores aos outputs que estavam, por sua vez, ligados diretamente à entrada da Placa Central. Deste modo, o sinal do contador de década pode ser utilizado como pin central destes transístores e assim os leds ignoram por completo a passagem de tensão pelos restantes componentes e podem ligar-se com toda a corrente e tensão que lhe é fornecida pela Placa Central. Paara além disso, agora todos os Leds estão com uma ligação a GROUND comum o que torna possível utilização de menos cabos entre componentes, sendo este o circuito final dos LEDs.

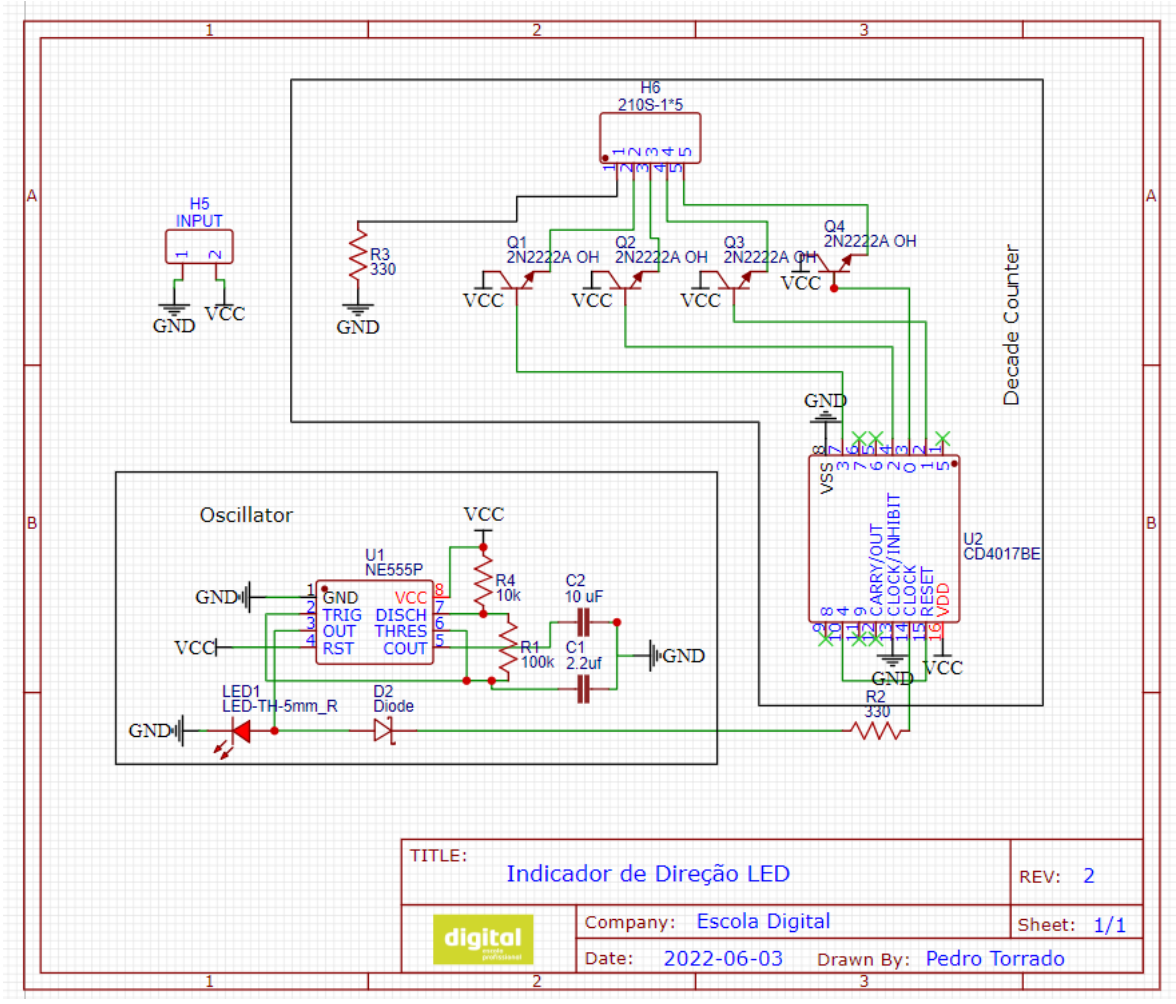

Figura 15 - Circuito Piscas Final

8. Programa

A programação destes circuitos foi desenvolvida ao mesmo tempo que os circuitos foram desenhados, mas após ter explicado e demonstrado a organização do circuito, os pinos do microprocessador a ser utilizado e os componentes em questão, será mais fácil imaginar a forma como o mesmo funciona. Para fazer este programa utilizei o editor de texto *Atom* como IDE utilizando *PlatformIO* para conseguir fazer upload de código e utilizar o Monitor de porta Serial. Estes ficheiros encontram-se na pasta Código. Tendo isso em consideração:

8.1 Comando

```
#include <Arduino.h>
#include <SoftwareSerial.h>

SoftwareSerial bt(2, 3);  //RX, TX

#define right_hes 5
#define left_hes 4
#define bt_enable 9
#define left_led 8
#define right_led 7

int left_state = 1;
int right_state = 1;
int verification_code;
```

Figura 16 - Bloco 1 de código comando (linha 27 a 40)

Como qualquer código de C++, o código começa por incluir as bibliotecas que vai utilizar, uma vez que estamos a utilizar um processador ATMEGA, de forma a podermos utilizar as suas capacidades da mesma forma que um arduino, temos que incluir a biblioteca *Arduino*, para além disso é importante incluirmos a biblioteca *SoftwareSerial* uma vez que com esta podemos passar os pinos RX (recetor) e TX (transmissor) para pinos diferentes dos próprios. Para isso, é importante não utilizar os pinos RX e TX integrados no microprocessador uma vez que se tivermos estes conectados a um componente, não será possível programar o microprocessador. Para definir quais são os pinos que pretendemos utilizar podemos verificar no nosso circuito, estes estão conectados ao módulo HC05 e são o pino 2 para recetor e 3 para o transmissor, isto pode tornar-se um pouco confuso uma vez que têm que ser o oposto daqueles no módulo Bluetooth, mas é fácil de perceber ao sabermos para que servem estes pinos. O pino TX ou transmissor, como o nome indica envia um sinal, portanto este tem que estar ligado a um pino RX ou recetor de forma a receber esse mesmo sinal. Assim, o RX do microcontrolador tem que estar conectado a TX do HC05 e RX do microcontrolador tem que estar conectado a TX do HC05. Portanto, apesar da primeira reação poder ser ligar os pinos

com os mesmos nomes, percebendo o conceito rapidamente percebemos a razão pela qual isso não está certo. Agora, para definirmos no código quais são esses pinos, utilizamos a função *SoftwareSerial* (da biblioteca com o mesmo nome) e definimos o nome do dispositivo que estamos a atribuir neste caso *bt()* dentro dessa função indicamos os pinos como se pode verificar no excerto do programa.

Após esta fase, são definidos os pinos para os sensores de campo eletromagnético (*right_hes*. *left_hes*), outro conectado à saída *STATE* do HC05 (*bt_enable*), e dois para leds com o objetivo de demonstrar a verificação de sinal enviado para a placa central (*left_led*, right_led). De seguida são definidas 3 variáveis globais de forma a que funcionem em qualquer função do programa, sendo essas *left_state* e *right_state* que têm como objetivo definir em que estado se encontram os leds *left_led* e *right_led*. Para além disso, é necessário criar a variável *verification_code* para usar mais tarde.

```
void setup() {
  //Bluetooth RX and TX defined
  pinMode(2, INPUT);
  pinMode(3, OUTPUT);

  //HC06 enable pin set to high
  pinMode(9, OUTPUT);
  digitalWrite(bt_enable, HIGH);
  bt.begin(9600);

  //Hall Effect Sensors defined
  pinMode(right_hes, INPUT);
  pinMode(left_hes, INPUT);

  //Verification LEDs defined
  pinMode(left_led, OUTPUT);
  pinMode(right_led, OUTPUT);

  Serial.begin(9600);
  Serial.println("Start Cntrlr");

  Serial.println("Pairing...");
}
```

Figura 17 - Bloco 2 de código comando (linha 42 a 64)

Depois do primeiro bloco de código, temos a função *setup()* sendo esta a função chamada no início de um programa e a forma standard de configurar o modo dos pinos. Ao longo desta configurei então cada um dos pinos necessários para os o funcionamento correto do programa, e isso é feito utilizando *pinMode()* e dentro desta indicamos o pino a que nos queremos referir e após isso se este é INPUT ou OUTPUT sendo INPUT a entrada de um sinal e OUTPUT a saída de sinal, ou de corrente. Para além disso, inicia o Monitor Serial a partir de *Serial.begin(9600)* sendo 9600 o baud rate do microprocessador e para que assim possamos verificar os processos em questão e facilitar o diagnóstico de problemas. De seguida podemos indicar ao utilizador por Monitor Serial que o programa está a funcionar como pretendido, apresentando essa informação no Monitor Serial como este programa faz com "Start Cntrlr" e "Pairing...", ao vermos isso podemos verificar não só que o microprocessador está a funcionar, mas que o dispositivo ainda se está a conectar (naturalmente uma vez que acabou de se ligar) e uma vez que este estiver conectado irá apresentar "Device Paired" , como veremos.

```
void verification(){

  if (bt.available()>0) {
    verification_code = bt.read();
    Serial.println(verification_code); //Shows the byte sent from the slave for trou
bleshooting

    if(verification_code == 10){
      digitalWrite(left_led, HIGH);
      //Shows what type of confirmation it has recieved, for troubleshooting in case
 of led problems
      Serial.println("left_confirmed_high");
    }
    else if(verification_code == 11){
      digitalWrite(left_led, LOW);
      Serial.println("left_confirmed_low");
    }
    else if(verification_code == 20){
      digitalWrite(right_led, HIGH);
      Serial.println("right_confirmed_high");
    }
    else if(verification_code == 21){
      digitalWrite(right_led, LOW);
      Serial.println("right_confirmed_low");
    }
    else if(verification_code == 91){
      Serial.println("Device Paired");
    }

    else{
      Serial.println("Input error : Not intended result or verification");
    }
  }
  else{
    verification_code = "";
  }
}
```

Figura 18 - Bloco 3 de código comando (linha 66 a 100)

De forma a ser possível a verificação de sinal para a qual era importante a comunicação a duas vias, que referi múltiplas vezes no capítulo 5 (Funcionalidades do Dispositivo), criei a função verification() cujo propósito é exatamente demonstrar uma verificação de sinal através dos LEDs conectados aos pinos de verificação (*left_led* e *right_led*). Mas como se faz?

No início da função, o programa verifica se houve alguma alteração de sinal no tipo de comandos enviados pelo outro dispositivo, isto é feito a partir da função *bt.available()* que apenas demonstra valores maiores de 0 quando um sinal é enviado quando não é, envia 0. Após fazer essa verificação iguala-se o valor enviado pelo outro dispositivo (lido com *bt.read()*) à variável *verification_code*,

que tinha sido criada no primeiro bloco de código. Se não foi enviado nenhum sinal, esta variável é igualada a “” ou seja, a nenhum valor inteiro.
Tendo esta variável, o programa verifica se é igual a um número de um conjunto de números, cada um com um significado.

CÓDIGO	PISCA	ESTADO
10	Esquerdo	Ligado
11	Esquerdo	Desligado
20	Direito	Ligado
21	Direito	Desligado
91	Nenhum	Dispositivos Conectados

Figura 19 - Tabela de Códigos de Verificação

A verificação do resultado pretendido ao enviar-se um sinal (que vamos ver como é feito no próximo bloco) é assim feita apenas por receber um sinal do outro dispositivo ao reproduzir a ação produzida. E se o código que receber não for um dos pretendidos podemos verificar esse erro a partir do Monitor Serial com o erro:

```
"Input error : Not intended result or verification"
```

E quando o resultado é o pretendido pode verificar-se uma mensagem como:

```
"right_confirmed_high"
```

```
void loop() {

  if (bt.available()>0) {
    verification();
  }

  left_state = digitalRead(left_hes);
  right_state = digitalRead(right_hes);
  //Show the output in Serial for troubleshooting
  //Serial.println(right_state);
  if (Serial.available()){
    bt.write(Serial.read());
  }

  if (right_state == LOW) {
    Serial.println("Right");
    //Show the output in Serial for troubleshooting
    //Serial.println('2');
    bt.write('2');
    delay(200);
    verification();
    delay(800);
  }

  if (left_state == LOW) {
    Serial.println("Left");
    //Show the output in Serial for troubleshooting
    //Serial.println('1');
    bt.write('1');
    delay(200);
    verification();
    delay(800);
  }
}
```

Figura 20 - Bloco 4 de código comando (linhas 102 a 135)

Na função *loop()* ao haver um sinal por parte do outro dispositivo, a função *verification()* é chamada de forma a fazer a verificação como explicada antes. Após isso são utilizadas as variáveis left_state e *right_state* que são igualadas a *digitalRead(left_hes)* e *digitalRead(right_hes)* sendo estes os valores lidos pelos sensores SS443A, que ao apresentarem HIGH ou 1, sabemos que não estão a detetar um campo eletromagnético e quando apresentam LOW ou 0, sabemos que estão a detetar um campo eletromagnético. Ou seja, neste caso 0 ou LOW significa que o utilizador quer alterar o estado do pisca do mesmo lado e quando está em HIGH significa que o utilizador não o acionou.
Após esse processo existe em comentário: `//Serial.println(right_state);`
Este deve ser utilizado, retirando "//" para verificar o estado do sensor direito de forma a verificar que o microprocessador está a ler os valores de pelo menos um dos sensores.

É possível passar qualquer input feito no Monitor Serial com:

```
if (Serial.available()){
  bt.write(Serial.read());
}
```

Facilitando o diagnóstico de problemas, uma vez que assim é possível passar informação diretamente para o outro dispositivo, sem ser necessário estar a utilizar os sensores, nem ter os mesmos funcionais. Podendo deste modo verificar se um problema pode existir com o sensor ou com a transmissão de informação.
Sabendo como são definidas as variáveis *right_state* e *left_state* o programa tudo o que faz é verificar se estes se encontram no estado LOW e se assim for apresenta essa informação no Monitor Serial apresentando `"Right"` e `"Left"` e após isso transmite-se 2 ou 1 para a placa central tendo 2 o significado de direito e 1 o significado de esquerdo (que como podemos verificar na última tabela que foi apresentada, o 1º dígito mantém a mesma organização).
Enviando o sinal, o microprocessador interrompe os processos durante 200ms utilizando a função *delay()* dando tempo ao outro dispositivo de receber o sinal. Terminados os 200ms inicia a função de verificação para que possa demonstrar que o pisca pretendido se encontra ligado ou desligado conforme o pretendido.

E com isto tem-se o comando. Mas, para ter noção de alguns destes processos é preciso perceber também de onde vêm alguns dos valores que este recebe, como na verificação e o que acontece quando são recebidos os valores enviados pelo comando.

8.2 Placa Central

Esta placa, como já referido tem como objetivo ser o recetor entre os dois piscas e o comando, ou seja, tem como função receber o sinal do comando e controlar o estado dos piscas e se estes se encontram ligados ou desligados, para além disso tem também que enviar o sinal para o comando a verificar o estado dos piscas como referido anteriormente.

```
#include <Arduino.h>
#include <SoftwareSerial.h>
SoftwareSerial bt(3, 2);  //RX, TX

#define left_led 7
#define right_led 8

int i;
int left_state = 1;
int right_state = 1;
int lr_state = 0;
int new_left_state = 1;
int new_right_state = 1;
int new_lr_state = 0;
char serial_command;
```

Figura 21 - Bloco 1 de Código Placa Central (Linha 23 a 37)

Assim como no comando, o início do programa tem como objetivo incluir as bibliotecas necessárias e os pinos para os componentes, sendo esses componentes apenas os pinos aos quais se ligam os circuitos dos piscas e aos quais chamei left_led e right_led. Para além disso, é preciso notar as variáveis definidas aqui uma vez que estas serão explicadas ao longo do programa.

```
void setup() {
  pinMode(3, INPUT);
  pinMode(2, OUTPUT);

  pinMode(left_led, OUTPUT);  // sets the digital pin 7 as output
  pinMode(right_led, OUTPUT);  // sets the digital pin 8 as output

  Serial.begin(9600);
  bt.begin(9600);

  Serial.println("Start Blnkrs");
}
```

Figura 22 - Bloco 2 de Código Placa Central (Linha 39 a 50)

Na função *setup()* fazemos o mesmo que no comando e definimos se os pinos estão a ser utilizados como entradas ou saídas de sinal e inicia-se a utilização do Monitor Serial.

```
void leftstate(){
  if(left_state == 0){
    bt.write(10);
    Serial.println("Verification Code Sent: 10");
  }
  else if(left_state == 1){
    bt.write(11);
    Serial.println("Verification Code Sent: 11");
  }
  else {
    Serial.println("Input error : state was changed but the result isn't valid");
  }
}
```

Figura 23 - Bloco 3 de Código Placa Central (Linha 52 a 64)

De forma a controlar o estado dos piscas e se possam enviar os códigos de verificação corretos, foram criadas duas funções *leftstate()* e *rightstate()*. Na figura acima temos representada a função *leftstate()* que tudo o que faz é verificar qual é o estado do respetivo pisca e caso este esteja desligado, envia o código 10 (como verificámos na tabela de códigos de verificação significa LED Esquerdo Ligado), e caso esteja ligado envia o respetivo código de verificação. A função *rightstate()* faz exatamente o mesmo processo, mas de forma a controlar o lado direito.

```
void loop() {

  //include both functions for verification
  int leftstate();
  int rightstate();

  if(bt.available()>0 or Serial.available()){
```

Figura 24 - Bloco 4 de Código Placa Central (Linhas 82 a 86)

A função *loop()* começa por definir as duas funções referidas anteriormente e para além disso verifica se há algum sinal enviado pelo módulo Bluetooth ou algum sinal dado através de Monitor Serial. Caso haja um deles:

```
    char command = bt.read();
    serial_command = Serial.read();

    if (command != ""){
      Serial.println(command);
    }
    else if (serial_command != ""){
      Serial.println(serial_command);
    }
```

```
if(command == '9' or serial_command == '9'){
  Serial.println("Device Paired");
  bt.write(91); // Verification Code sent to Controller to verify connection
  leftstate();
  rightstate();
}
```

Figura 25 – Bloco 5 de Código Placa central (Linhas 88 a 103)

É criada uma variável do tipo caractere, chamada *command*, que é igualada ao sinal enviado por Bluetooth e a variável serial_command, igualada ao sinal dado por Monitor Serial. Se qualquer um dos dois enviar um sinal diferente de "" este é apresentado no Monitor Serial. Depois disso o comando ao receber o código 9 este envia o código 91 para que o comando saiba que se encontram conectados, testando assim também essa conexão.

```
if (command == '1' or serial_command == '1') {

  if (new_left_state == 1) {
    digitalWrite(left_led, HIGH);  // sets the digital pin 7 ON
    Serial.println("LEFT_HIGH");   // print on serial monitor the pin mode
    left_state = 0;
  }
  else if (new_left_state == 0) {
    digitalWrite(left_led, LOW);  // sets the digital pin 7 OFF
    Serial.println("LEFT_LOW");   // print on serial monitor the pin mode
    left_state = 1;
  }
  new_left_state = left_state;
  leftstate();
}
```

Figura 26 - Bloco 6 de Código Placa Central (Linhas 144 a 158)

Ao receber o código 0, 1 ou 2 o programa vai sempre fazer o mesmo processo alterando apenas as variáveis e dispositivos em questão. No exemplo acima está a ser utilizado o pisca esquerdo e de forma a evitar que o programa mude o estado do mesmo e envie sinais de verificação constantemente é necessário utilizar uma variável global que neste caso é *new_left_state* e que não se altera a não ser que o comando código 1 seja recebido, caso seja, assim como na função *leftstate()* este verifica o estado do led e caso este esteja ligado faz uma ação e caso esteja desligado faz outra, sendo que em vez de apenas enviar um sinal, este desliga e liga o led, respetivamente.

E assim, muito simplesmente, é possível alterar o estado dos piscas utilizando os sinais enviados pelo comando e é também possível enviar a informação necessária de forma que o mesmo possa demonstrar essa informação.

9. PCBs

Após desenvolver os desenhos dos circuitos, foi possível desenvolver os PCBs, seguindo as mesmas ligações e sendo o resultado final o seguinte:

9.1 Comando

O primeiro circuito a ser desenvolvido foi preparado para ser utilizado com uma fonte externa para testes e por isso foi feito sem um circuito de carregamento e proteção da bateria.
Ao desenvolver este PCB decidi também tentar passar o módulo Bluetooth para a camada inferior do PCB, de forma a ser possível diminuir as dimensões do mesmo, sendo o resultado o seguinte:

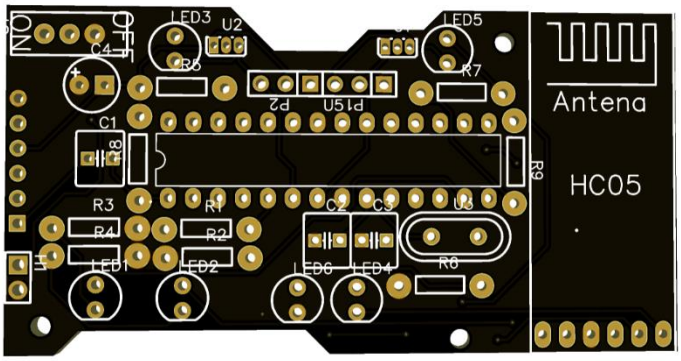

Figura 27 - PCB Comando V1.0

Após esta versão, decidi acrescentar um circuito para carregamento e proteção da bateria por indução magnética.

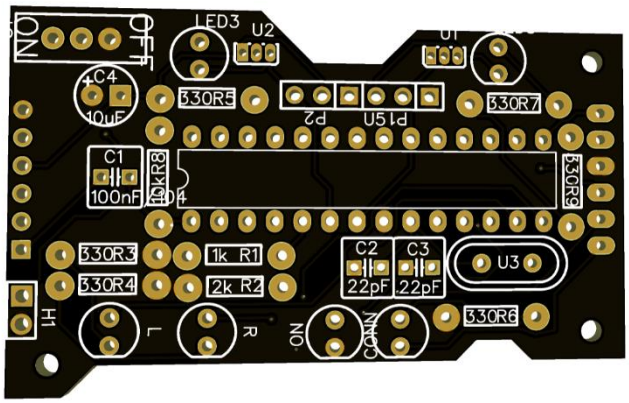

Figura 28 - PCB Comando V1.1 - Topo

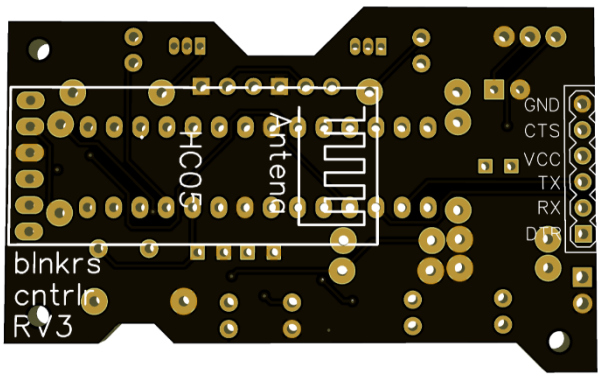

Figura 29 - PCB Comando V1.1 - Inferior

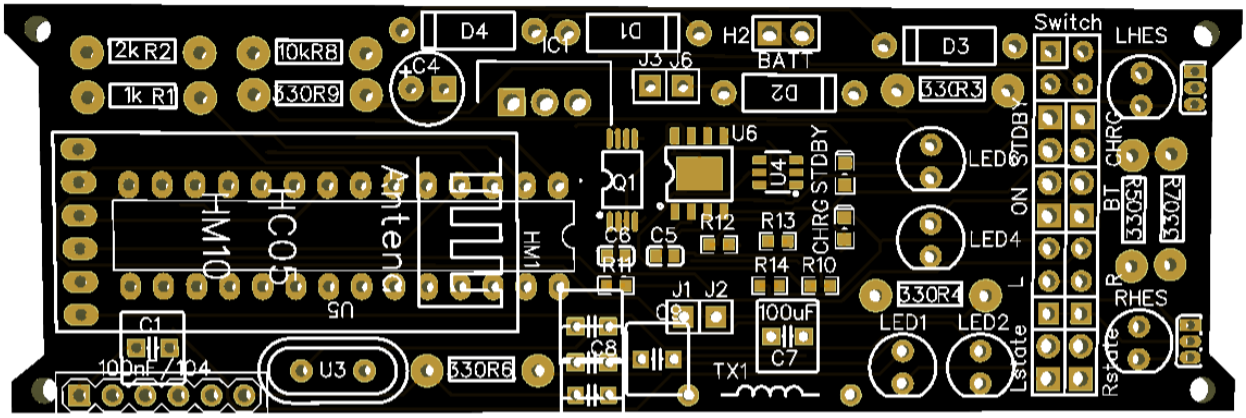

Figura 30 - PCB Comando V2.0

Ao desenvolver este PCB, decidi também organizar pinos de forma a ter acesso aos leds de forma externa, caso queira ligar os mesmos a outros LEDs externos. Apesar deste circuito ter bastantes mais funcionalidades que a primeira versão ele tem dimensões bastante maiores, e por isso decidi alterar vários dos componentes para SMD (Surface Mount Device) uma vez que estes componentes são bastante mais pequenos e assim pude também passar mais componentes para a parte inferior do PCB sem aumentar a altura do mesmo.

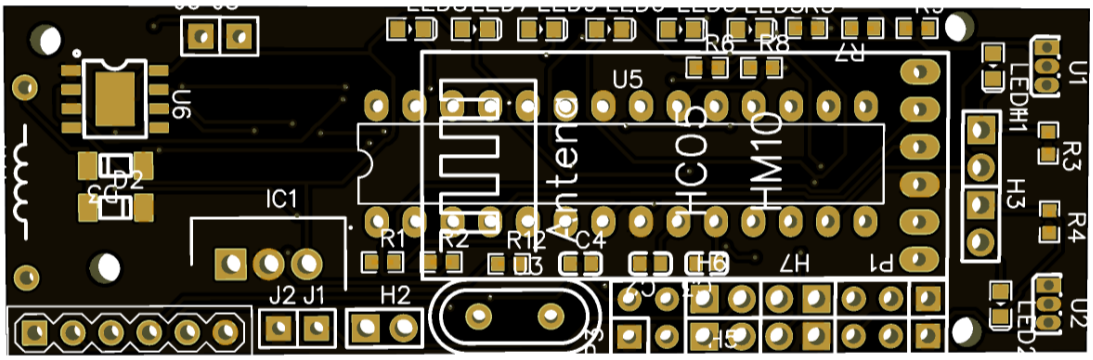

Figura 31 - PCB Comando V2.1 – Topo

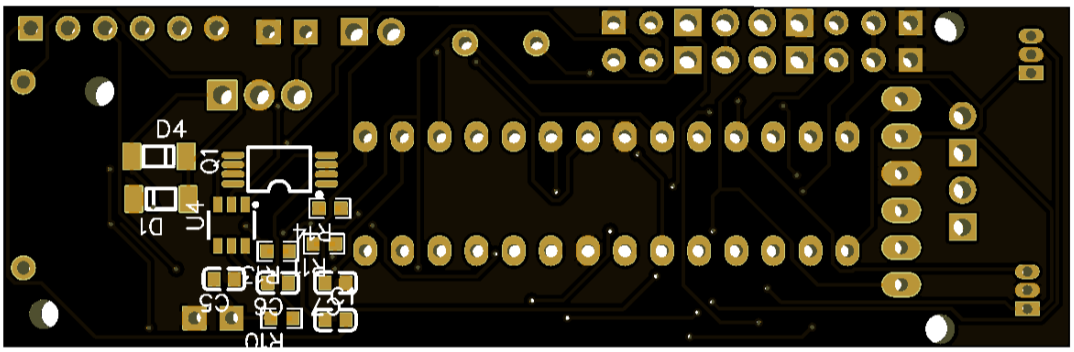

Figura 32 - PCB Comando V2.1 – Base

9.2 Placa Central

Ao desenvolver o circuito central tive menos preocupação com o tamanho do mesmo, uma vez que este não tem as mesmas limitações de espaço que o comando e por isso não cheguei a utilizar a camada inferior do circuito, nem a utilizar componentes SMD.
Assim como no comando, a primeira versão tinha sido preparada para uma fonte externa e por isso tive que fazer as mesmas alterações.

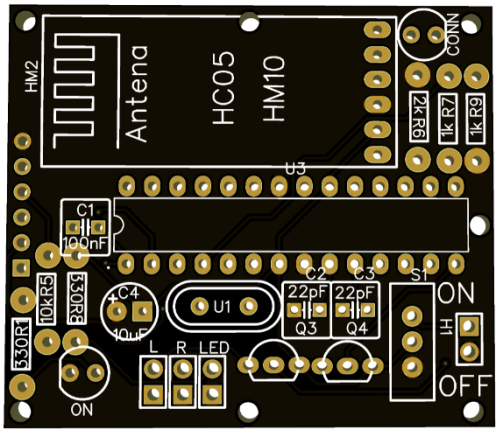

Figura 33 - PCB Placa Central V1

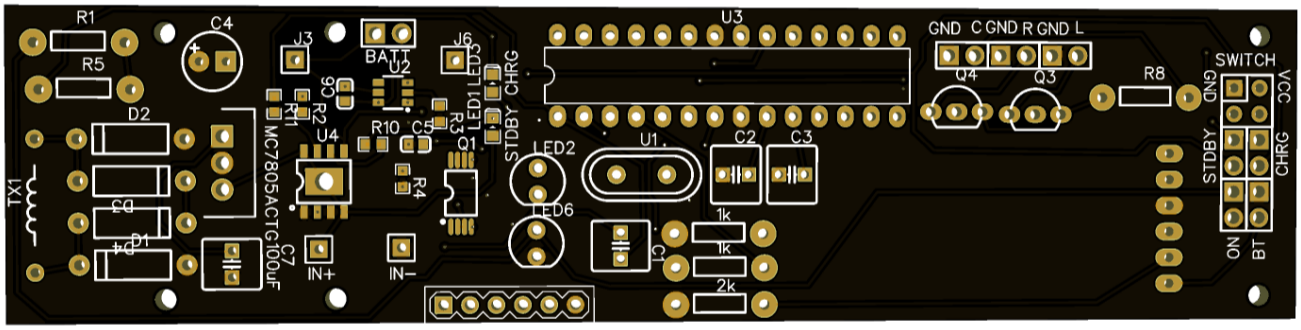

Figura 34 - PCB Placa Central V2

9.3 LEDs

As alterações entre estes dois modelos são a implantação de transístores e a transição de condensadores eletrolíticos para cerâmicos, uma vez que estes são têm menores dimensões e por isso será mais fácil implementá-los mais tarde nos dispositivos.

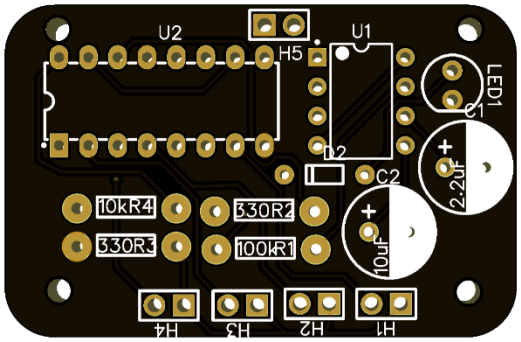

Figura 35 - PCB LEDs V1

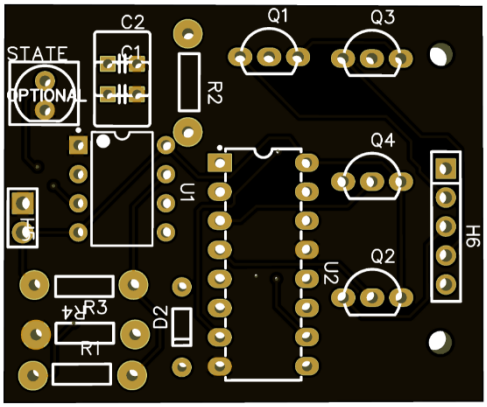

Figura 36 - PCB LEDs V2

10. Modelos 3D

Uma vez que o meu curso não se foca na área de CAD (Computer Assisted Design) não irei descrever os processos que passei para desenvolver as caixas que envolvem a continuação e execução deste projeto. Considerei importante realçar que foi necessário ter muitas tolerâncias em consideração durante o desenvolvimento destes modelos, uma vez que é necessário tornar estes os mais protegidos possível do exterior. O software que utilizei para produzir estes modelos foi o Fusion 360 da AutoDesk. As medidas exatas dos modelos finais apresentados abaixo podem ser consultados nos ficheiros com o respetivo nome na pasta "CAD Drawings" em anexo.

10.1 Comando

O primeiro modelo do comando foi desenhado para a primeira versão do PCB e para ser utilizado com uma bateria Li-Ion (Ião Lítio), o que afetou bastante o seu desenho, como se pode notar na imagem:

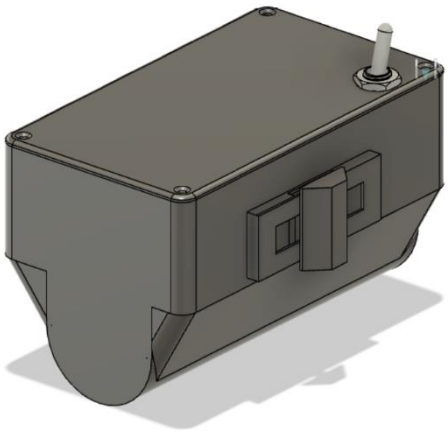

Figura 37 - CAD Comando V1, Dimensões(mm), x: 69.80, y: 39.95, z: 42.39

Quando atualizei para o segundo modelo do PCB foi necessário fazer muitas alterações, uma vez que as dimenões do PCB eram diferentes e o local onde se encontravam os componentes eram também diferentes. Tinha ainda que ter em consideração a inclusão de uma bobina de forma a poder carregar o dispositivo. O modelo anterior não possuía qualquer forma de carregar a bateria, uma vez que foi desenvolvido apenas como uma forma de conceptualizar a utilização de uma bateria Li-ion, como referi antes.
A segunda versão tinha como objetivo ser mais compacta e de fácil utilização.

Figura 38 - CAD Comando V2 Dimensões(mm), x: 69.80, y: 39.95, z: 42.39

Este modelo foi desenvolvido para o PCB versão 2.2 do comando e tem uma grande vantagem em relação à primeira versão, que é a inclusão de uma capa que seria produzida num material mais flexível e que oferece proteção ao dispositivo, o material usado para projetar este tipo de utilização foi TPU, sendo o resto do componente desenvolvido para material como PLA, sendo PLA bom para fazer estruturas sólidas e TPU bom exatamente para o tipo de estrutura mais flexível que pretendo alcançar.
Este modelo, ao contrário do anterior, incluí uma bobina de forma a ser possível carregar por indução eletromagnética, e esta bobina está posicionada como se pode ver na seguinte figura:

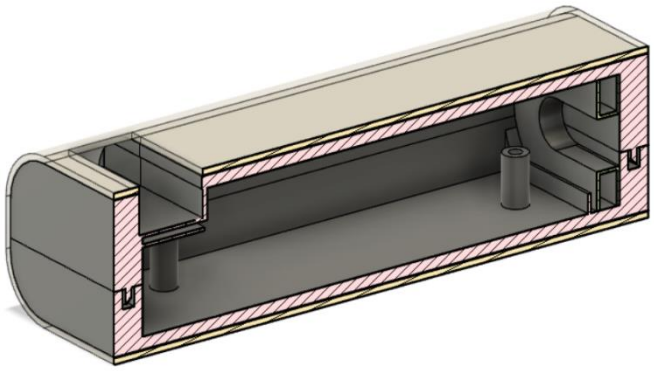

Figura 39 - CAD Comando V2 Secção

Na figura 39 temos o mesmo modelo da figura 38, mas seccionado de forma a podemos ver como é feito o encaixe e onde se encontra a bobinha, sendo esta o componente mais à direita do interior do modelo. Contudo, esta tem um problema que me obrigou a mudar completamente o modelo em questão. Não era possível fazer a bobina grande o suficiente para que esta funcionasse de forma correta com a bobina do transmissor. A única forma de fazer a bobina grande o suficiente era alterá-la para uma forma cilíndrica como apresentado na figura, mas tal causa muitos problemas, uma vez que não é ótima para indução eletromagnética (devendo-se utilizar formas mais circulares).

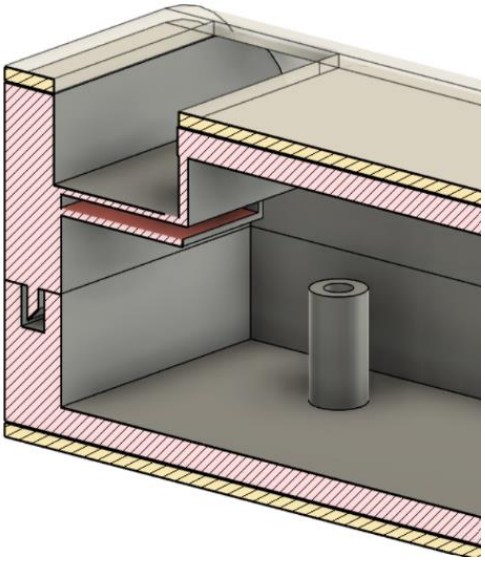

Figura 40 - CAD Comando V2 Secção 2

Para visualizar o mecanismo que funciona com os sensores eletromagnéticos alterei a visão da figura 38 para a direita. Este mecanismo utiliza 2 ímanes de neodímio com dimensões de 10mm por 10mm e com uma altura de 1mm. Escolhi estes ímanes especialmente pelo seu tamanho pequeno. Um dos ímanes é colocado no espaço cuja base está marcada a vermelho. Acima deste espaço deve ser colocado outro íman com as mesmas características que ao mover-se para qualquer um dos lados deve voltar a ser colocado alinhado com o centro, sendo alinhado por força magnética com o íman que se encontra dentro da caixa, sendo assim possível certificarmo-nos de que o íman nunca fica a ser lido continuamente por nenhum dos sensores, a não ser que o utilizador o faça intencionalmente.
O último modelo desenhado foi desenhado com o objetivo de ser mais pequeno que o anterior e que corrigisse todos os problemas apresentados anteriormente.

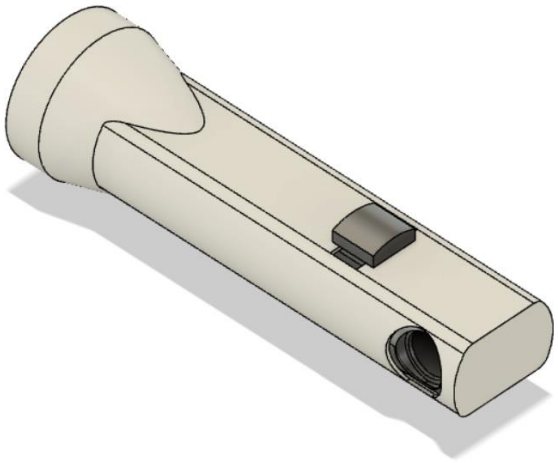

Figura 41 - CAD Comando V3 Dimensões (mm), x: 122.43, y: 38, z: 19.60

Este modelo criado para resolver o maior problema do anterior, tem uma zona que foi desenhada exatamente com o tamanho da bobina em mente, daí a zona cónica e cilíndrica maior do dispositivo. A bobina encontra-se no dispositivo, como podemos ver nas imagens acima, e tendo um espaço que se encontra vazio (quando o PCB ainda não tiver sido colocado ou tiver sido retirado) é possível deslizar a bobina para o sítio pretendido.

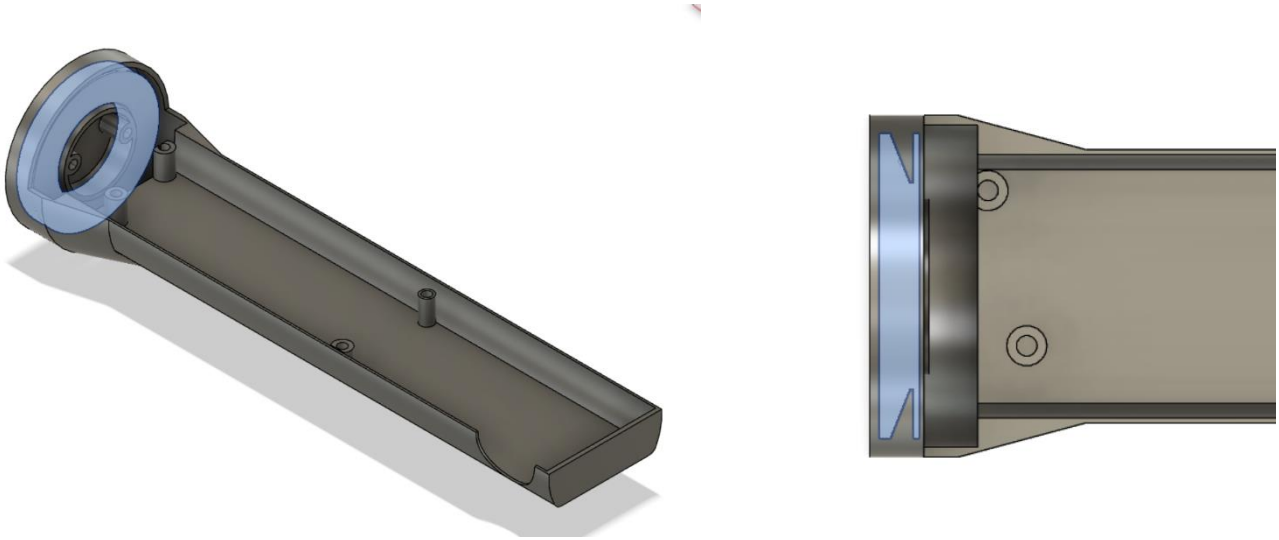

Figura 42 - CAD Comando V3 Base, visão isométrica e superior

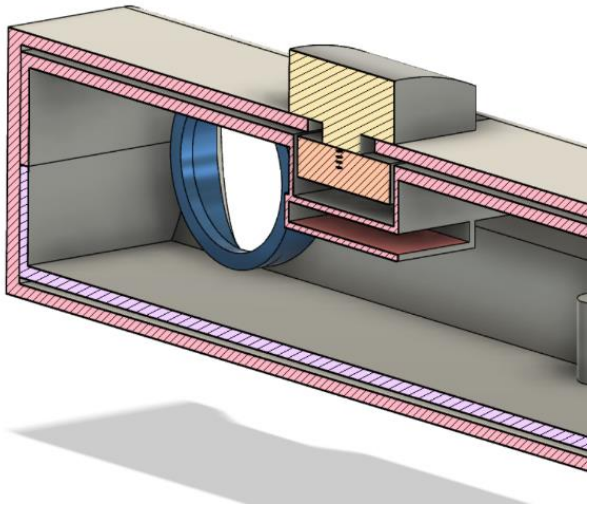

Figura 43 - CAD Comando V3 Secção

Nesta figura podemos verificar que o mecanismo envolvendo os ímanes mantém-se parecido, contudo foi adicionada uma peça com o propósito de facilitar o movimento do íman e, assim, a utilização do dispositivo. Este mecanismo, sendo composto por duas partes, é possível montar deixando metade abaixo da proteção e, sendo esta metade maior que o espaço aberto na capa, este fica preso no componente sem interferir com a sua liberdade de movimento. Para além disso podemos verificar que se encontra uma parte em azul neste desenho, e essa parte é um buraco desenhado especificamente para um botão (Componente 21), com mecanismo interruptor, com as dimensões apresentadas na figura 45.

Este botão possui um led interno que torna possível demonstrar se o dispositivo se encontra ligado ou não sem ser necessário uma LED dedicado a isso.

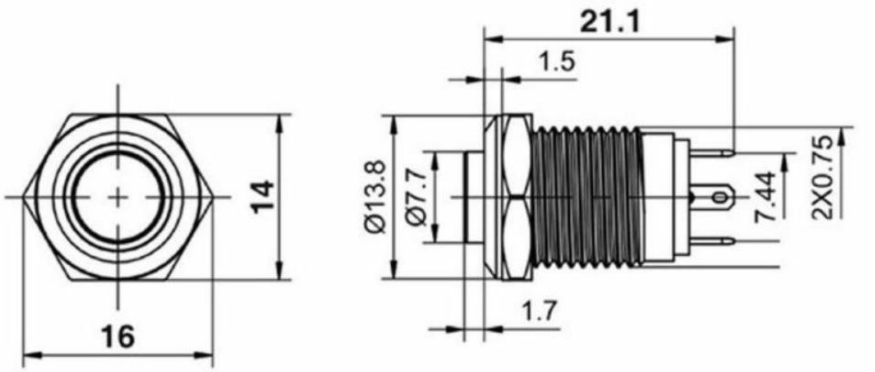

Figura 44 - Dimensões Botão Interruptor

10.2 Placa Central

Este modelo tinha bastante menos limitações no seu desenvolvimento que o comando, sendo que no seu desenho não havia preocupações com ergonomia, por isso o primeiro modelo também já incluía mais capacidades.

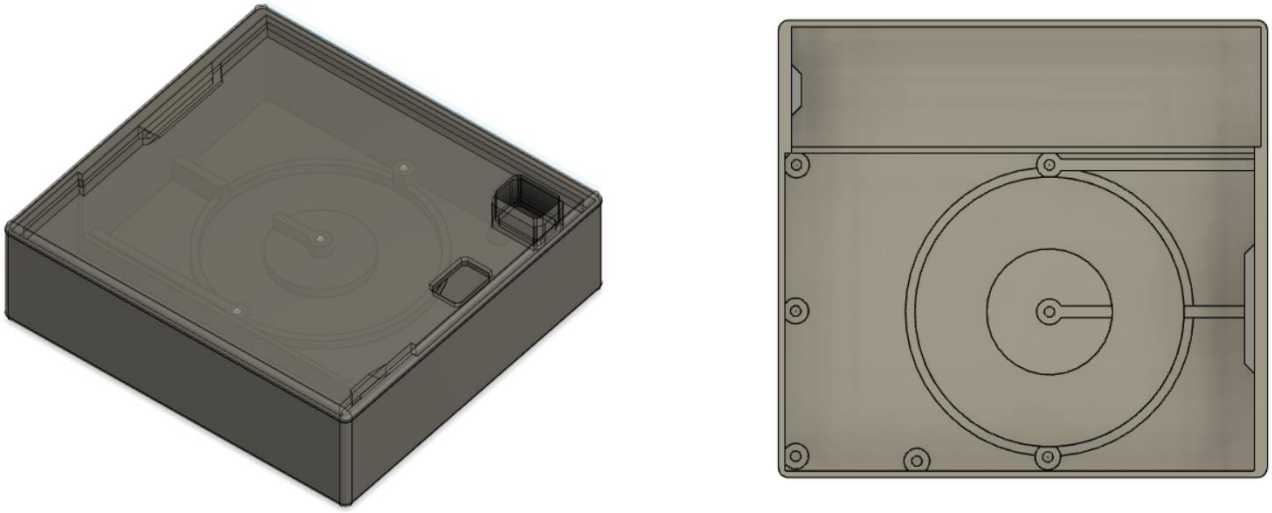

Figura 45 - CAD Placa Central V1 Dimensões(mm), x: 70.75, y: 75.78, z: 23.00, visão isométrica e superior

Este incluía já um espaço para uma bobina na sua base como se pode ver nas figuras acima, este utilizava um módulo de indução eletromagnética (Componente 22), sendo que a versão seguinte já tem a capacidade deste módulo implementada no circuito e por isso ocupa também bastante menos espaço. Este módulo estava organizado como se apresenta na figura 46.

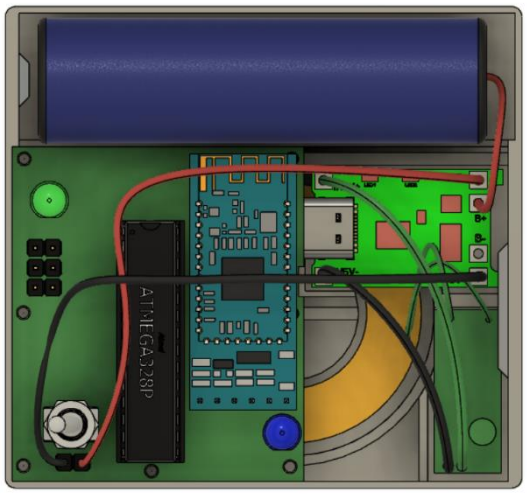

Figura 46 - CAD Placa Central V1 Organizada

Após o desenvolvimento desta placa era necessário implementar as novas capacidades da segunda versão da placa de circuito impresso, e, também, torná-lo mais durador.

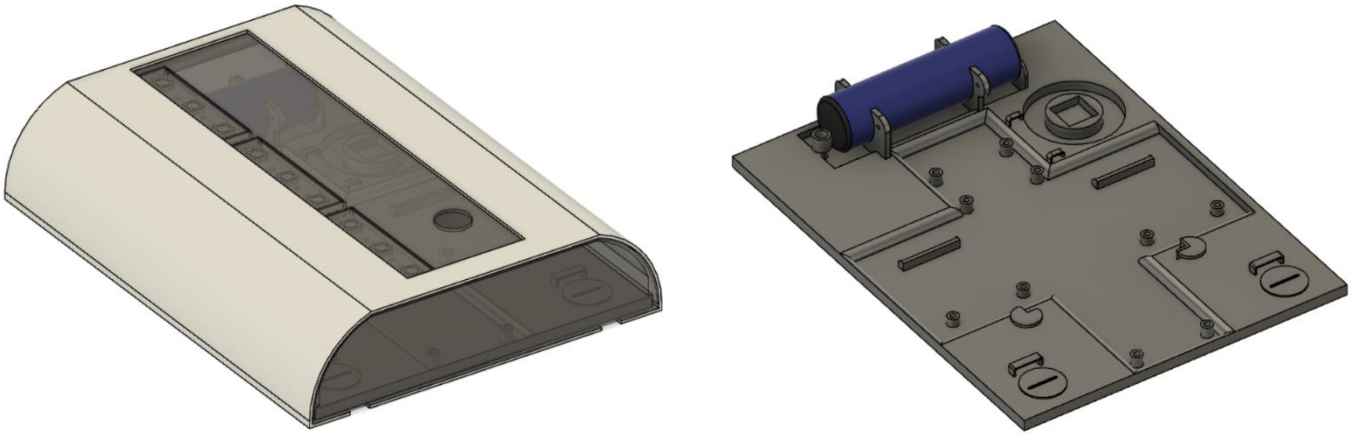

Figura 47 - CAD Placa Central V2 Dimensões(mm), x: 70.75, y: 75.78, z: 23.00, visão isométrica com e sem topo e capa

Nesta versão podemos verificar que houve uma grande alteração e aumentou bastante em tamanho mas mantendo-se dentro do portátil, uma vez que facilita a interação com o utilizador e mantém o seu ponto de equilíbrio no centro, o que é importante para a sua função. Para além disso, em comparação com a primeira versão, este inclui os circuitos dos piscas diretamente, mantendo-os fáceis de substituir caso haja algum problema de hardware e também mantendo esses componentes mais protegidos e permitindo maior flexibilidade no design dos Piscas. Como é fácil de verificar este também possui uma capa, pela mesma razão que os comandos e no seu topo terá o botão 3 fitas LED com o propósito de transmitir a informação necessária ao utilizador quando preparar o dispositivo e sendo também uma forma de chamar atenção a outros condutores quando em utilização, mesmo que os piscas se encontrem desligados.

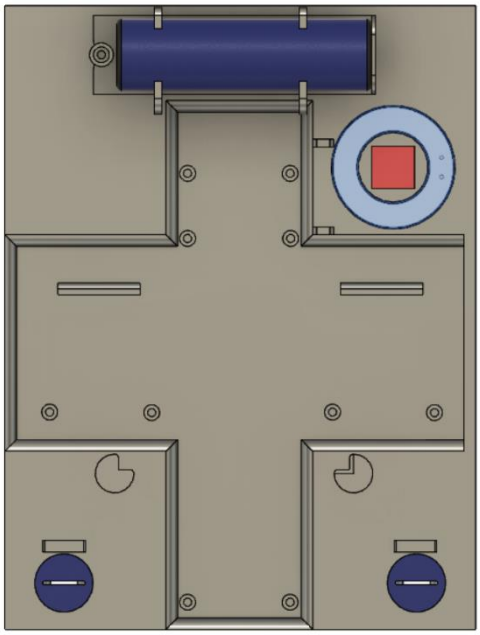

Figura 49 - CAD Placa Central V2 Topo

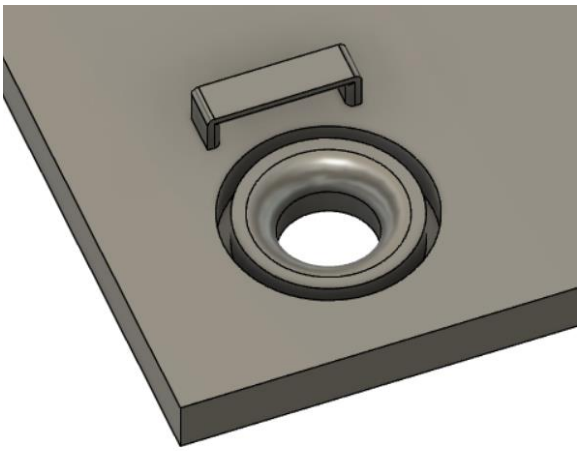

Figura 48 - CAD Placa Central V2 Saída de Cabos

Algumas das capacidades deste modelo inclui uma capa para a saída de cabos pela parte inferior do dispositivo com capas representadas a azul-escuro na figura (não confundir com a bateria) estas capas tornam possível passar cabos para conectar aos piscas, sem haver ranhuras expostas para entrada de líquidos (à prova de água). Para além disso podemos verificar onde se encontra a bobina a azul-claro e que no seu centro existe um espaço para um íman igual aos dos comandos, que pode permitir ao carregador alinhar-se sozinho com o sítio pretendido sem ser necessário mecanismos complexos.

10.3 Piscas

O modelo que tem como objetivo incluir os LEDs, que demonstraram a função realizada pelo circuito que apresentei anteriormente com o nome "LEDs", assim como qualquer um dos anteriores modelos, também passou por mais que uma fase, sendo a primeira a seguinte:

Figura 50 - Piscas V1, Dimensões (mm) x: 80.02,y: 47.20, z: 6.50

Este foi desenhado para ser utilizado sempre em conjunto com outro idêntico, um de cada lado do primeiro modelo da Placa Central, e como se pode verificar é composto por 4 espaços, cada um com um tamanho diferente, um para cada fita, e para cada saída do circuito. Estes utilizavam LEDs 5050 (cujo modelo e dimensões podem ser verificados no respetivo ficheiro em "CAD Drawings").
Após desenvolver este primeiro modelo, houve vários problemas que se apresentaram, sendo um deles que, apesar de ser possível transmitir a informação necessária a qualquer condutor que se encontrasse atrás do utilizador, era impossível informar os condutores à frente através deste sistema, por isso foi necessário alterar o mesmo. Assim, de forma que o objeto pudesse ser utilizado de forma mais visível este poderia ser utilizada nos ombros do utilizador, e com essa ideia desenvolvi o seguinte modelo:

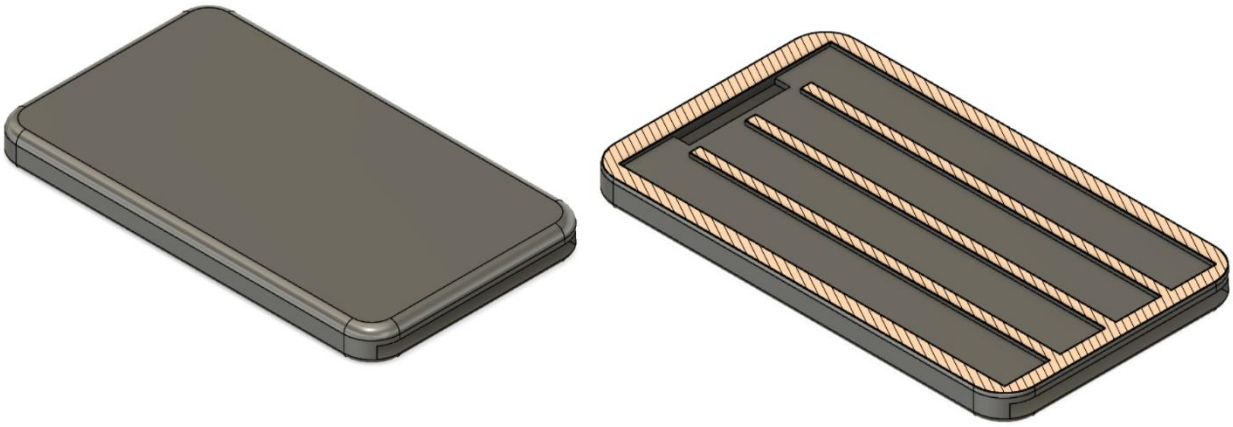

Figura 51 - Piscas V2

Este modelo, ao contrário do interior, foi desenhado para material flexível. Uma vez que estando nos ombro este precisa de tomar a forma dos mesmos e para além disso tem uma melhor capacidade de resistência, uma vez que este terá na sua composição apenas fitas LED, não há problema em dobrar os mesmos pois estas fitas são também flexiveis. Para além disso, e para tentar isolar estes o melhor possível do exterior, estes modelos são impressos todos de uma vez e na camada anterior à primeira camada do topo do modelo, são colocados os LEDs (D100) no seu respetivo local, com as ligações necessárias para o correto funcionamento de todos.
Finalmente, e para ser possível colocar estes nos ombros e a Placa Central às costas é necessário que estes estejam suportados no nosso corpo de alguma forma. Para isso foi comprada uma fita de 3 metros, com 38,5mm de largura e 1,5mm de espessura.

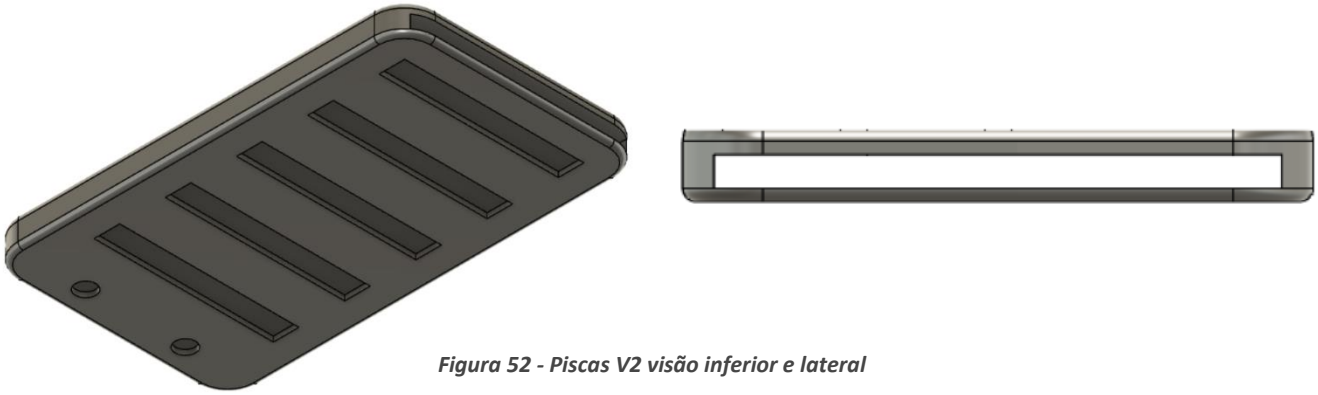

Figura 52 - Piscas V2 visão inferior e lateral

Esta fita é colocada no espaço entre a base deste objeto e o sítio onde estão colocados os LEDs, sendo assim possível ajustar a posição dos mesmos.

10.4 Fivela

De forma a prender então o dispositivo ao corpo é preciso criar algum mecanismo para que este se possa encontrar seguro. Para isso foram desenvolvidas 3 peças diferentes, sendo duas delas a própria fivela da fita que prende o dispositivo.

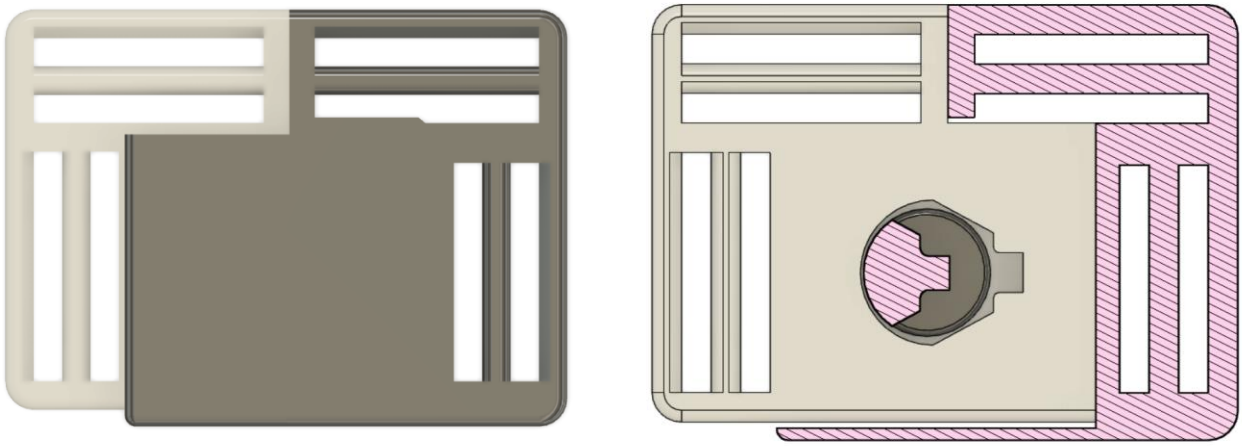

Figura 53 – Fivela Dimensões (mm) x: 99.00,y: 73.50, z: 10.00

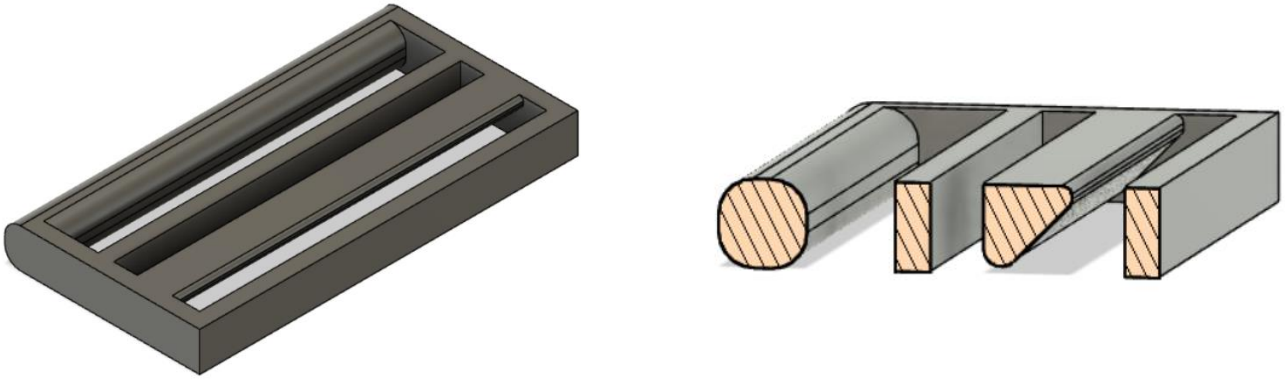

Figura 54 - Peça Anti-Deslize x: 48.50, y: 25.00, z: 5.00

Acima podemos verificar o funcionamento do mesmo sendo que cada uma das peças está representada em cores diferentes de forma a ser mais fácil de entender o propósito de cada. E em cada um dos lados e em cima podemos verificar como fica presa e apertada a fita a estes compoentes sendo necessários 4, uma vez que a fita da uma volta à cintura (por cada lado) e ainda dá uma volta por cima dos ombros (onde estão presos os Piscas) dando assim uma grande liberdade de ajuste ao utilizador. Para além disso foram preparadas 4 peças, uma para cada uma das entradas da fita de forma que seja possível apertar e soltar a mesma sem haver fita extra pendurada.

Cada ponta da fita pode ser assim presa à parte redonda da fita e a partir da forma como está desenhada esta peça, como se pode identificar na visão por seção, pelo ângulo em que entra em contacto com a fita esta fica presa e não desliza sobre a mesma, ficando assim presa e ajustada ao tamanho pretendido pelo utilizador.

De forma a prender a Placa Central, a parte de baixo da capa foi desenhada da seguinte forma. Mais uma vez, a visão à direita é o mesmo modelo mas seccionado, sendo assim possível verificar o espaço que foi deixado para que seja possível passar uma fita horizontal e prender 2 fitas diagonais cada uma destinada a um ombro. Para além disso foi deixado espaço na capa para a passagem dos cabos para os piscas e ainda um circulo na zona do carregador sem fios evitando assim interferencias desnecessárias por acresentar camadas entre as duas bobinas.

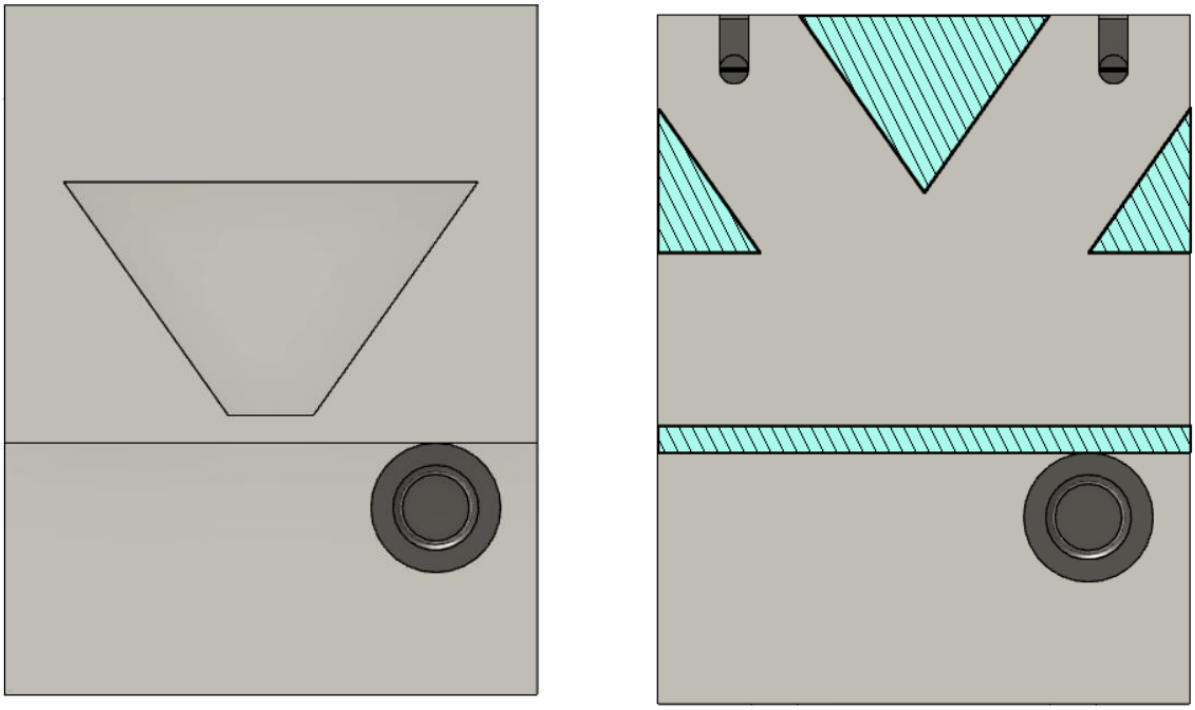

Figura 55 - Capa Placa Central

11. Implementação

Este produto pode ser implementado de muitas formas e em diversos contextos. O mais óbvio é o uso pessoal e compra individual do mesmo. Contudo, este projeto pode ser utilizado de outras formas, como por exemplo implementação do mesmo diretamente noutros dispositivos, como incluir os circuitos em skateboards elétricos (podendo incluir as funcionalidades do comando no comando do próprio skateboard). O exemplo abaixo demonstra uma possível implementação de um mecanismo como o comando deste projeto, mas implementado num comando modelo de um skateboard elétrico, sem alterar nenhuma das outras funcionalidades do comando.

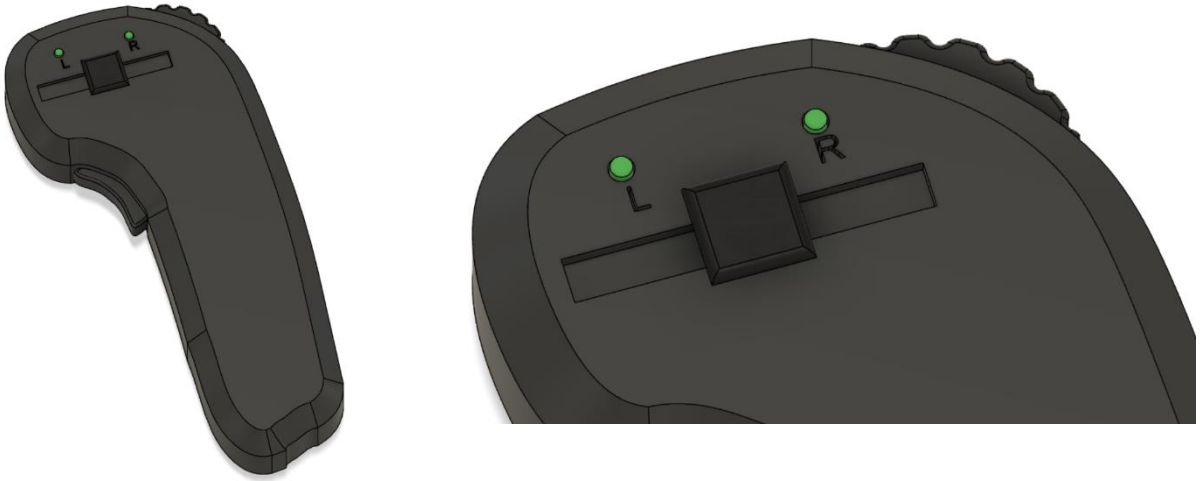

Figura 56 - Exemplo de Implementação skateboard elétrico.

Uma implementação como esta não seria difícil, uma vez que estes comandos já funcionam sem fios com o resto do dispositivo e assim a única parte do circuito a implementar são os sensores eletromagnéticos (ou outro tipo de mecanismo com o mesmo propósito) e os LEDs que demonstram o envio e a receção correta do dispositivo.
Outro dispositivo no qual se poderia incluir um circuito como o deste projeto é nas trotinetes elétricas públicas ou individuais. Podendo nesse caso remover por completo a comunicação sem fios e podendo colocar os piscas em vários sítios da trotinete como se vê na figura 57:

Figura 57 - Exemplo de Implementação Trotinete

Neste caso não seria necessário implementar um circuito sem fios, uma vez que é possível ligar tudo diretamente através da estrutura da trotinete, o mesmo se aplicando para bicicletas caso este fosse implementado diretamente na mesma.
O negativo de qualquer uma destas implementações em comparação com a desenvolvida neste projeto é que estas são bastante menos versáteis e são utilizáveis apenas com o próprio método de transporte para o qual é desenvolvido, enquanto o **Line Light** foi desenvolvido com o propósito de ser um dispositivo utilizável com qualquer um destes meios de locomoção.
Contudo, é possível desenvolvê-lo das formas apresentadas acima para uso em transportes de uso partilhado (como as trotinetes públicas que vemos praticamente em qualquer local de grande movimentação) ou caso seja do interesse de um produto no qual seja apropriado utilizar as várias funcionalidades que apresentei ao longo deste relatório.

12. Componentes

1. Módulos Radiofrequência – https://www.ptrobotics.com/315mhz-e-433mhz/4798-433mhz-transmitter-and-receiver-module.html?gclid=CjwKCAjwj42UBhAAEiwACIhADqthTRPae9yi9HpzSbxvlWugX3BNToj2-SAwIzNRKYVXv4APOPe4vhoCIYYQAvD_BwE
2. Recetor Infravermelho – https://www.ptrobotics.com/sensores-infravermelho/8756-tsop38540-receptor-de-infravermelhos-40khz.html
3. LED infravermelho – https://www.ptrobotics.com/sensores-infravermelho/319-tsus5400-950nm-infrared-led.html
4. Módulo HC-05 – https://www.ptrobotics.com/modulos-bluetooth/4364-modulo-bluetooth-hc-05-para-arduino.html
5. Módulo HC-06 – https://www.ptrobotics.com/modulos-bluetooth/4794-bluetooth-module-hc-06.html
6. BreadBoard – https://www.ptrobotics.com/breadboards/792-breadboard-830-pontos-transparente.html
7. Arduino Nano – https://www.ptrobotics.com/arduino/5678-arduino-nano-v30-compativel-com-cabo-usb-.html
8. Cabo USB 2.0 mini B – Incluído com Arduíno UNO
9. Carregador de baterias de lítio tp4056 – https://www.ptrobotics.com/alimentacao/8449-modulo-de-carregamento-de-baterias-com-proteccao-de-carga-tp4056-usb-c.html
10. Baterias de lítio mr18650 (3.7v e 2200mAh) – https://mauser.pt/catalog/product_info.php?products_id=120-0445
11. Cabos de conexão – https://www.ptrobotics.com/jumper-wires/6482-premium-male-male-jumper-wires-100mm-pack-of-40.html?gclid=CjwKCAjwj42UBhAAEiwACIhADrTGe4WEkE_8hFJ4JAsbaxh4UKvPXpY4zIgSyty_kIy1aKV9iafSdhoCjgMQAvD_BwE
12. Arduíno UNO – https://www.ptrobotics.com/arduino/5542-arduino-uno-r3-compatible-ch340-chip-w-usb-cable.html
13. FTDI FT232RL USB - https://www.ptrobotics.com/conversores/6988-ftdi-ft232rl-usb-to-ttl-serial-converter-adapter-module-5v-and-33v.html?gclid=CjwKCAjwj42UBhAAEiwACIhADuxKo3j1_eZRdYrwFQjDwdDDpx492QFXgIKCjEYdvzi24xeeyaZv4xoC2OkQAvD_BwE
14. THT LEDs – https://www.ptrobotics.com/kits-led/5704-ptrobotics-5mm-led-500pcs-kit-vermelho-blue-verde-amarelo-and-branco.html
15. Transistors PNP – https://www.electrofun.pt/componentes-eletronicos/kit-100-transistores-diversos-velleman?utm_campaign=efshopping&utm_source=google&utm_medium=cpc&utm_source=google&utm_medium=shopping&utm_campaign=roas&gclid=CjwKCAjwj42UBhAAEiwACIhADpoHvcm-5GIC2IQczF0whgkx5yM7FuJ0pp3T1rr9sIxiIVCUIPplORoCwnEQAvD_BwE
16. Knight Rider – https://www.ptrobotics.com/electronica-essencial/4836-knight-rider-eletronica-essencial-.html

17. Condensadores – https://www.ptrobotics.com/kits-componentes/10331-kit-de-condensadores-ceramicos-2pf-a-100nf-300pcs.html
18. Resistências – https://www.ptrobotics.com/kit-de-resistencias/1006-kit-resistencias-ptrobotics.html
19. Díodos – https://www.ptrobotics.com/diodos-schottky/3399-bat42-schottky-diode-200ma-30v.html
20. a3144 – www.shorturl.at/qxyB3
21. Botão Interruptor - https://www.ebay.com/itm/401567861377
22. Módulo Indução Eletromagnética - https://mauser.pt/catalog/product_info.php?cPath=1667_2604_1758&products_id=096-0294

13. Ficheiros em Anexo

Anexo 1 - Utilização de sinais luminosos em velocípedes, trotinetes, skateboards e outros veículos. (Respostas).pdf

Anexo 2 - Calendarização.xlsx

Anexo 3 - BOM.xlsx

Circuitos:

Anexo 4 – LineLight_Comando.pdf
Anexo 5 – LineLight_LEDs.pdf
Anexo 6 – LineLight_Placa_Principal.pdf

Anexo 7 – Encomendas.xlsx

Código:

Anexo 8 – Comando.cpp
Anexo 9 – Placa_Central.cpp

PCBs:

Anexo 10 – Gerber_PCB_blnkrs_Controller_RV2.1
Anexo 11 – Gerber_PCB_blnkrs_MainBoard_RV2.0
Anexo 12 – Gerber_PCB_Controller_RV1.0
Anexo 13 – Gerber_PCB_Controller_RV1.1
Anexo 14 – Gerber_PCB_Controller_RV2.0
Anexo 15 – Gerber_PCB_LEDs_RV1
Anexo 16 – Gerber_PCB_LEDs_RV2
Anexo 17 – Gerber_PCB_MainBoard_RV1.0

CAD Drawings:

Anexo 18 – Bobina.pdf
Anexo 19 – Comando.pdf
Anexo 20 – LED_D100.pdf
Anexo 21 – LED_RGB.pdf
Anexo 22 – Placa_Central.pdf
Anexo 23 – Fivela.pdf
Anexo 24 – Fastner.pdf

14. Índice de Imagens

Figura 1 - Objetivos de Desenvolvimento Sustentável 4
Figura 2 - Cronograma 5
Figura 3 - Sistema de verificação 6
Figura 4 - ATMEGA328P Comando 10
Figura 5 - Circuito de SS443A e HC-05 11
Figura 6 - Circuito de Carregamento 12
Figura 7 - Corrente Alternada (Azul), Sinal Retificado (Vermelho), Corrente Contínua (Verde) 12
Figura 8 - Circuito Comando 13
Figura 9 - NE555P 14
Figura 10 - CD4017BE 14
Figura 11 - gráfico de output do circuito integrado NE555P 15
Figura 12 - Circuito de Contador de Década e Output do sistema 15
Figura 13 - Circuito completo de Indicador Luminoso de Direção 16
Figura 14 - Circuito com função de Oscilador 16
Figura 15 - Circuito Piscas Final 17
Figura 16 - Bloco 1 de código comando (linha 27 a 40) 18
Figura 17 - Bloco 2 de código comando (linha 42 a 64) 20
Figura 18 - Bloco 3 de código comando (linha 66 a 100) 21
Figura 19 - Tabela de Códigos de Verificação 22
Figura 20 - Bloco 4 de código comando (linhas 102 a 135) 23
Figura 21 - Bloco 1 de Código Placa Central (Linha 23 a 37) 25
Figura 22 - Bloco 2 de Código Placa Central (Linha 39 a 50) 25
Figura 23 - Bloco 3 de Código Placa Central (Linha 52 a 64) 26
Figura 24 - Bloco 4 de Código Placa Central (Linhas 82 a 86) 26
Figura 25 – Bloco 5 de Código Placa central (Linhas 88 a 103) 27
Figura 26 - Bloco 6 de Código Placa Central (Linhas 144 a 158) 27
Figura 27 - PCB Comando V1.0 28
Figura 28 - PCB Comando V1.1 - Topo 28
Figura 29 - PCB Comando V1.1 - Inferior 28
Figura 30 - PCB Comando V2.0 29
Figura 31 - PCB Comando V2.1 – Topo 29
Figura 32 - PCB Comando V2.1 – Base 29

Figura 33 - PCB Placa Central V1 30
Figura 34 - PCB Placa Central V2 30
Figura 35 - PCB LEDs V1..... 31
Figura 36 - PCB LEDs V2..... 31
Figura 37 - CAD Comando V1, Dimensões(mm), x: 69.80, y: 39.95, z: 42.39..... 32
Figura 38 - CAD Comando V2 Dimensões(mm), x: 69.80, y: 39.95, z: 42.39..... 33
Figura 39 - CAD Comando V2 Secção 33
Figura 40 - CAD Comando V2 Secção 2 34
Figura 41 - CAD Comando V3 Dimensões (mm), x: 122.43, y: 38, z: 19.60..... 35
Figura 42 - CAD Comando V3 Base, visão isométrica e superior 35
Figura 43 - CAD Comando V3 Secção..... 36
Figura 44 - Dimensões Botão Interruptor 36
Figura 45 - CAD Placa Central V1 Dimensões(mm), x: 70.75, y: 75.78, z: 23.00, visão isométrica e superior..... 37
Figura 46 - CAD Placa Central V1 Organizada 37
Figura 47 - CAD Placa Central V2 Dimensões(mm), x: 70.75, y: 75.78, z: 23.00, visão isométrica com e sem topo e capa..... 38
Figura 48 - CAD Placa Central V2 Saída de Cabos..... 39
Figura 49 - CAD Placa Central V2 Topo 39
Figura 50 - Piscas V1, Dimensões (mm) x: 80.02,y: 47.20, z: 6.50..... 40
Figura 51 - Piscas V2 41
Figura 52 - Piscas V2 visão inferior e lateral 41
Figura 53 – Fivela Dimensões (mm) x: 99.00,y: 73.50, z: 10.00 42
Figura 54 - Peça Anti-Deslize x: 48.50, y: 25.00, z: 5.00..... 42
Figura 55 - Capa Placa Central..... 43
Figura 56 - Exemplo de Implementação skateboard elétrico. 44
Figura 57 - Exemplo de Implementação Trotinete..... 45